

(19) 日本国特許庁(JP)

(12) 特許公報(B2)

(11) 特許番号

特許第6728240号
(P6728240)

(45) 発行日 令和2年7月22日(2020.7.22)

(24) 登録日 令和2年7月3日(2020.7.3)

(51) Int.Cl.	F 1
HO4N 19/13 (2014.01)	HO4N 19/13
HO4N 19/157 (2014.01)	HO4N 19/157
HO4N 19/174 (2014.01)	HO4N 19/174
HO4N 19/70 (2014.01)	HO4N 19/70

請求項の数 13 (全 66 頁)

(21) 出願番号	特願2017-561642 (P2017-561642)	(73) 特許権者	595020643 クアアルコム・インコーポレイテッド QUALCOMM INCORPORATED アメリカ合衆国、カリフォルニア州 92121-1714、サン・ディエゴ、モアハウス・ドライブ 5775
(86) (22) 出願日	平成28年5月27日(2016.5.27)	(74) 代理人	100108855 弁理士 蔵田 昌俊
(65) 公表番号	特表2018-521555 (P2018-521555A)	(74) 代理人	100109830 弁理士 福原 淑弘
(43) 公表日	平成30年8月2日(2018.8.2)	(74) 代理人	100158805 弁理士 井関 守三
(86) 國際出願番号	PCT/US2016/034695	(74) 代理人	100112807 弁理士 岡田 貴志
(87) 國際公開番号	W02016/196307		
(87) 國際公開日	平成28年12月8日(2016.12.8)		
審査請求日	令和1年5月13日(2019.5.13)		
(31) 優先権主張番号	62/168,503		
(32) 優先日	平成27年5月29日(2015.5.29)		
(33) 優先権主張国・地域又は機関	米国(US)		
(31) 優先権主張番号	15/166,068		
(32) 優先日	平成28年5月26日(2016.5.26)		
(33) 優先権主張国・地域又は機関	米国(US)		

最終頁に続く

(54) 【発明の名称】高度算術コーダ

(57) 【特許請求の範囲】

【請求項 1】

ビデオデータをエントロピーコーディングする方法であって、前記方法は、前記ビデオデータのスライス中のシンタックス要素のための値をエントロピーコーディングするために、コンテキスト適応型エントロピーコーディングプロセスにおいて使用される複数のコンテキストのうちのコンテキストのためのあらかじめ定義された初期化値を取得することと、ここにおいて、前記あらかじめ定義された初期化値がNビット精度で記憶される。

前記あらかじめ定義された初期化値に基づいて、およびルックアップテーブルを使用して、前記ビデオデータの前記スライスのための前記コンテキストの初期確率状態を決定することと、ここにおいて、前記コンテキストのための可能な確率状態の数が2のN乗よりも大きく、前記ルックアップテーブル中のi番目の値は、

前記コンテキストのための可能な確率状態の前記数と、1シンボルの可能な確率のセットを表すテーブル中のi番目の値との積と、

オフセット値と

の和のc e i lに等しい整数である、

前記コンテキストの前記初期確率状態に基づいて、前記シンタックス要素のための前記値のビンをエントロピーコーディングすることとを備える、方法。

【請求項 2】

前記オフセット値は、第1のオフセット値であり、前記方法は、

10

20

前記あらかじめ定義された初期化値に基づいて、傾斜値と第2のオフセット値とを決定すること

をさらに備え、ここにおいて、前記初期確率状態を決定することは、

前記傾斜値と、前記第2のオフセット値と、前記ビデオデータの前記スライスの量子化パラメータとに基づいて中間値を決定することと、

前記ルックアップテーブルを使用して、前記中間値を前記初期確率状態にマッピングすることと

を備える、請求項1に記載の方法。

【請求項3】

前記ルックアップテーブルは、2のN乗よりも小さいかまたはそれに等しい数のエントリを含む、請求項2に記載の方法。 10

【請求項4】

可能な確率のセットを表す前記テーブル中の前記i番目の値が1シンボルの前記i番目の可能な確率である、請求項1に記載の方法。

【請求項5】

前記コンテキストのための可能な確率状態の前記数は、2のN乗よりも大きいかまたはそれに等しい、請求項1に記載の方法。

【請求項6】

Nが8であり、前記コンテキストのための可能な確率状態の前記数が2の15乗である、請求項4に記載の方法。 20

【請求項7】

前記オフセット値が0.5または0に等しい、請求項1に記載の方法。

【請求項8】

エントロピーコーディングすることは、前記ビンをエントロピー符号化することを備える、請求項1に記載の方法。

【請求項9】

エントロピーコーディングすることは、前記ビンをエントロピー復号することを備える、請求項1に記載の方法。

【請求項10】

前記コンテキスト適応型エントロピーコーディングプロセスは、コンテキスト適応型バイナリ算術コーディング(CABAC)プロセス、またはコンテキスト適応型可変長コーディング(CAVLC)プロセスを備える、請求項1に記載の方法。 30

【請求項11】

前記ルックアップテーブルは、以下の式

【数1】

```

m_MappedProb[128] =
{
  614, 647, 681, 718, 756, 797, 839, 884, 932, 982, 1034, 1089, 1148, 1209, 1274, 1342,
  1414, 1490, 1569, 1653, 1742, 1835, 1933, 2037, 2146, 2261, 2382, 2509, 2643, 2785,
  2934, 3091, 3256, 3430, 3614, 3807, 4011, 4225, 4452, 4690, 4941, 5205, 5483, 5777,
  6086, 6412, 6755, 7116, 7497, 7898, 8320, 8766, 9235, 9729, 10249, 10798, 11375,
  11984, 12625, 13300, 14012, 14762, 15551, 16384, 16384, 17216, 18005, 18755, 10
  19467, 20142, 20783, 21392, 21969, 22518, 23038, 23532, 24001, 24447, 24869,
  25270, 25651, 26012, 26355, 26681, 26990, 27284, 27562, 27826, 28077, 28315,
  28542, 28756, 28960, 29153, 29337, 29511, 29676, 29833, 29982, 30124, 30258,
  30385, 30506, 30621, 30730, 30834, 30932, 31025, 31114, 31198, 31277, 31353,
  31425, 31493, 31558, 31619, 31678, 31733, 31785, 31835, 31883, 31928, 31970,
  32011, 32049, 32086, 32120, 32153,
}.

```

20

を備える、請求項1に記載の方法。

【請求項12】

ビデオデータのエントロピーコーディングのための装置であって、前記装置は、

前記ビデオデータのスライス中のシンタックス要素のための値をエントロピーコーディングするために、コンテキスト適応型エントロピーコーディングプロセスにおいて使用される複数のコンテキストのうちのコンテキストのためのあらかじめ定義された初期化値を取得するための手段と、ここにおいて、前記あらかじめ定義された初期化値がNビット精度で記憶される、

前記あらかじめ定義された初期化値に基づいて、およびルックアップテーブルを使用して、前記ビデオデータの前記スライスのための前記コンテキストの初期確率状態を決定するための手段と、ここにおいて、前記コンテキストのための可能な確率状態の数が2のN乗よりも大きく、前記ルックアップテーブル中のi番目の値は、

前記コンテキストのための可能な確率状態の前記数と、1シンボルの可能な確率のセットを表すテーブル中のi番目の値との積と、

オフセット値と

の和のc_e_i_lに等しい整数である、

前記コンテキストの前記初期確率状態に基づいて、前記シンタックス要素のための前記値のビンをエントロピーコーディングするための手段と

を備える、装置。

【請求項13】

実行されたとき、ビデオコーディングデバイスの1つまたは複数のプロセッサに、請求項1乃至1_1のいずれかに記載の方法を行わせる命令を記憶するコンピュータ可読記憶媒体。

40

【発明の詳細な説明】

【技術分野】

【0001】

[0001] 本出願は、その内容全体が参照により本明細書に組み込まれる、2015年5月29日に出願された米国仮出願第62/168,503号の利益を主張する。

【0002】

[0002] 本開示は、ビデオコーディング(video coding)に関し、より詳細には、ビデ

50

オデータ (video data) のバイナリ算術コーディング (binary arithmetic coding) のための技法に関する。

【背景技術】

【0003】

[0003] デジタルビデオ機能は、デジタルテレビジョン、デジタルダイレクトブロードキャストシステム、ワイヤレスブロードキャストシステム、携帯情報端末 (PDA) 、ラップトップまたはデスクトップコンピュータ、デジタルカメラ、デジタル記録デバイス、デジタルメディアプレーヤ、ビデオゲームデバイス、ビデオゲームコンソール、セルラーまたは衛星無線電話、ビデオ遠隔会議デバイスなどを含む、広範囲にわたるデバイスに組み込まれ得る。デジタルビデオデバイスは、デジタルビデオ情報をより効率的に送信、受信および記憶する (store) ための、MPEG-2、MPEG-4、ITU-T H.263、ITU-T H.264 / MPEG-4, Part 10, アドバンストビデオコーディング (AVC : Advanced Video Coding) によって定義された規格、高効率ビデオコーディング (HEVC : High Efficiency Video Coding) 規格、およびそのような規格の拡張に記載されているビデオ圧縮技法など、ビデオ圧縮技法を実装する。

【0004】

[0004] ビデオ圧縮技法は、ビデオシーケンスに固有の冗長性を低減または除去するための空間予測および / または時間予測を含む。ブロックベースのビデオコーディングの場合、ビデオフレームまたはスライス (slice) はブロックに区分され得る。各ブロックはさらに区分され得る。イントラコード化 (I) フレームまたはスライス中のブロックは、同じフレームまたはスライス中の隣接ブロック中の参照サンプルに対する空間予測を使用して符号化される。インターフォーマット (P または B) フレームまたはスライス中のブロックは、同じフレームまたはスライス中の隣接ブロック中の参照サンプルに対する時間予測、あるいは他の参照フレーム中の参照サンプルに対する時間予測を使用し得る。空間予測または時間予測は、コーディングされるべきブロックのための予測ブロックを生じる。残差データ (Residual data) は、コーディングされるべき元のブロックと予測ブロックとの間のピクセル差分を表す。

【0005】

[0005] インターコード化ブロック (inter-coded block) は、予測ブロックを形成する参照サンプルのブロックを指す動きベクトル (motion vector) と、コード化ブロックと予測ブロックとの間の差分 (difference) を示す残差データとに従って符号化される。イントラコード化ブロックは、イントラコーディングモードと残差データとに従って符号化される。さらなる圧縮のために、残差データは、ピクセル領域から変換領域に変換され、残差変換係数 (residual transform coefficient) が生じ得、その残差変換係数は、次いで量子化され得る。最初は 2 次元アレイに構成される量子化された変換係数は、エントロピーコーディング (entropy coding) のための変換係数の 1 次元ベクトルを生成するために、特定の順序で走査され得る。

【0006】

[0006] 様々なエントロピーコーディングプロセスが、残差変換係数、動きベクトル情報、シンタックス要素 (syntax element) 、および他の関連する情報をコーディングするために利用得る。様々なエントロピーコーディングおよび他のデータ圧縮プロセスの例としては、コンテキスト適応型可変長コーディング (CAVL : context-adaptive variable length coding) 、コンテキスト適応型バイナリ算術コーディング (CABAC : context-adaptive binary arithmetic coding) 、確率間隔区分エントロピーコーディング (PIPE : probability interval partitioning entropy coding) 、ゴロムコーディング (Golomb coding) 、ゴロム - ライスコーディング (Golomb-Rice coding) 、および指数ゴロムコーディング (exponential Golomb coding) がある。

【発明の概要】

【0007】

[0007] 概して、本開示は、ビデオコーディングを実行するための技法について説明す

10

20

30

40

50

る。より詳細には、本開示は、異なるウィンドウサイズで C A B A C を実行するための例示的な技法について説明する。

【0008】

[0008] 一例では、ビデオデータのエントロピーコーディングのための方法は、ビデオデータのスライス中のシンタックス要素のための値 (value) をエントロピーコーディングするために、コンテキスト適応型コーディングプロセスにおいて使用される複数のコンテキスト (context) のうちの 1 つのコンテキストのためのあらかじめ定義された初期化値 (pre-defined initialization value) を取得する (obtain) ことと、ここにおいて、あらかじめ定義された初期化値が N ビット精度 (N-bit precision) で記憶される、およびあらかじめ定義された初期化値に基づいて、ビデオデータのスライスのためのコンテキストの初期確率状態 (initial probability state) を決定する (determine) ことと、ここにおいて、コンテキストのための可能な確率状態の数 (a number of possible probability states for the context) が 2 の N 乗 (two raised to the power of N) よりも大きい、コンテキストの初期確率状態に基づいて、シンタックス要素のための値のビン (bin) をエントロピーコーディングすることとを含む。この例では、本方法はまた、コーディングされたビンに基づいてコンテキストの確率状態 (probability state) を更新することと、コンテキストの更新された確率状態に基づいて、同じコンテキストをもつ次のビンをエントロピーコーディングすることとを含む。10

【0009】

[0009] 別の例では、ビデオデータのエントロピーコーディングのための装置 (apparatus) は、ビデオデータのスライス中のシンタックス要素のための値をエントロピーコーディングするために、コンテキスト適応型エントロピーコーディングプロセス (context-adaptive entropy coding process) において使用される複数のコンテキストを記憶するように構成されたメモリ (memory) と、1 つまたは複数のプロセッサ (processor) とを含む。この例では、1 つまたは複数のプロセッサは、複数のコンテキストのうちのコンテキストのためのあらかじめ定義された初期化値を取得することと、ここにおいて、あらかじめ定義された初期化値が N ビット精度で記憶される、あらかじめ定義された初期化値に基づいて、ビデオデータのスライスのためのコンテキストの初期確率状態を決定することと、ここにおいて、コンテキストのための可能な確率状態の数が 2 の N 乗よりも大きい、コンテキストの初期確率状態に基づいて、シンタックス要素のための値のビンをエントロピーコーディングすることとを行うように構成される。この例では、1 つまたは複数のプロセッサは、コーディングされたビンに基づいてコンテキストの確率状態を更新することと、コンテキストの更新された確率状態に基づいて、同じコンテキストをもつ次のビンをエントロピーコーディングすることとを行うようにさらに構成される。2030

【0010】

[0010] 別の例では、ビデオデータのエントロピーコーディングのための装置は、ビデオデータのスライス中のシンタックス要素のための値をエントロピーコーディングするために、コンテキスト適応型エントロピーコーディングプロセスにおいて使用される複数のコンテキストのうちのコンテキストのためのあらかじめ定義された初期化値を取得するための手段と、ここにおいて、あらかじめ定義された初期化値が N ビット精度で記憶される、あらかじめ定義された初期化値に基づいて、ビデオデータのスライスのためのコンテキストの初期確率状態を決定するための手段と、ここにおいて、コンテキストのための可能な確率状態の数が 2 の N 乗よりも大きい、コンテキストの初期確率状態に基づいて、シンタックス要素のための値のビンをエントロピーコーディングするための手段とを含む。この例では、本装置はまた、コーディングされたビンに基づいてコンテキストの確率状態を更新するための手段と、コンテキストの更新された確率状態に基づいて、同じコンテキストをもつ次のビンをエントロピーコーディングするための手段とを含む。40

【0011】

[0011] 別の例では、コンピュータ可読記憶媒体 (computer-readable storage medium) は、実行されたとき、ビデオコーディングデバイスの 1 つまたは複数のプロセッサに、50

ビデオデータのスライス中のシンタックス要素のための値をエントロピーコーディングするために、コンテキスト適応型エントロピーコーディングプロセスにおいて使用される複数のコンテキストのうちのコンテキストのためのあらかじめ定義された初期化値を取得することと、ここにおいて、あらかじめ定義された初期化値がNビット精度で記憶される、あらかじめ定義された初期化値に基づいて、ビデオデータのスライスのためのコンテキストの初期確率状態を決定することと、ここにおいて、コンテキストのための可能な確率状態の数が2のN乗よりも大きい、コンテキストの初期確率状態に基づいて、シンタックス要素のための値のビンをエントロピーコーディングすることを行わせる命令(*instruction*)を記憶する。この例では、コンピュータ可読記憶媒体はまた、1つまたは複数のプロセッサに、コーディングされたビンに基づいてコンテキストの確率状態を更新することと、コンテキストの更新された確率状態に基づいて、同じコンテキストをもつ次のビンをエントロピーコーディングすることを行わせる命令を記憶する。

【0012】

[0012] 別の例では、コンピュータ可読記憶媒体は、ビデオ復号デバイス(video decoding device)によって処理されたとき、ビデオ復号デバイスの1つまたは複数のプロセッサに、ビデオデータのスライス中のシンタックス要素のための値をエントロピーコーディングするために、コンテキスト適応型エントロピーコーディングプロセスにおいて使用される複数のコンテキストのうちのコンテキストのためのあらかじめ定義された初期化値を取得することと、ここにおいて、あらかじめ定義された初期化値がNビット精度で記憶される、あらかじめ定義された初期化値に基づいて、ビデオデータのスライスのためのコンテキストの初期確率状態を決定することと、ここにおいて、コンテキストのための可能な確率状態の数が2のN乗よりも大きい、コンテキストの初期確率状態に基づいて、シンタックス要素のための値のビンをエントロピーコーディングすることを行わせる符号化ビデオデータを記憶する。この例では、コンピュータ可読記憶媒体はまた、1つまたは複数のプロセッサに、コーディングされたビンに基づいてコンテキストの確率状態を更新することと、コンテキストの更新された確率状態に基づいて、同じコンテキストをもつ次のビンをエントロピーコーディングすることを行わせる命令を記憶する。

【0013】

[0013] 本開示の1つまたは複数の態様の詳細が添付の図面および以下の説明に記載されている。本開示で説明される技法の他の特徴、目的、および利点は、その説明および図面、ならびに特許請求の範囲から明らかになろう。

【図面の簡単な説明】

【0014】

【図1】[0014] 例示的なビデオ符号化および復号システム(video encoding and decoding system)を示すブロック図。

【図2A】[0015] バイナリ算術コーディングにおける範囲更新プロセス(range update process)を示す概念図。

【図2B】バイナリ算術コーディングにおける範囲更新プロセスを示す概念図。

【図3】[0016] バイナリ算術コーディングにおける出力プロセスを示す概念図。

【図4】[0017] 例示的なビデオエンコーダ(video encoder)を示すブロック図。

【図5】[0018] ビデオエンコーダ中のコンテキスト適応型バイナリ算術コーダ(context adaptive binary arithmetic coder)を示すブロック図。

【図6】[0019] 例示的なビデオデコーダ(video decoder)を示すブロック図。

【図7】[0020] ビデオデコーダ中のコンテキスト適応型バイナリ算術コーダを示すブロック図。

【図8】[0021] 通常コーディングモード(regular coding mode)を使用する所与のビン値のためのバイナリ算術符号化プロセス(binary arithmetic encoding process)を示す図。

【図9】[0022] 残差4分木(residual quadtree)に基づく例示的な変換方式を示す概念図。

10

20

30

40

50

【図10】[0023] 係数グループ (coefficient group) に基づく例示的な係数走査 (coefficient scan) を示す概念図。

【図11】[0024] 本開示の1つまたは複数の技法による、異なるウィンドウサイズでコンテキストベースエントロピー符号化 (context-based entropy encoding) を実行するための例示的なプロセスを示すフローチャート。

【図12】[0025] 本開示の1つまたは複数の技法による、異なるウィンドウサイズでコンテキストベースエントロピー復号 (context-based entropy decoding) を実行するための例示的なプロセスを示すフローチャート。

【発明を実施するための形態】

【0015】

[0026] 本開示の技法は、概して、ブロックベースハイブリッドビデオコーディング (block-based hybrid video coding) におけるエントロピーコーディングモジュールに関する。これらの技法は、HEVC (高効率ビデオコーディング (High Efficiency Video Coding)) など、任意の既存のビデオコーデック (video codec) に適用され得、あるいはこれらの技法は、任意の将来のビデオコーディング規格または他のプロプライエタリ (proprietary) もしくは非プロプライエタリコーディング技法における効率的なコーディングツールであり得る。例および説明のために、本開示の技法は、概して、HEVC (またはITU-T H.265) および/またはITU-T H.264に関して説明される。

【0016】

[0027] 図1は、可変ウィンドウサイズ (variable window size) でCABAC設計に従ってデータをコーディングするための技法を利用し得る例示的なビデオ符号化および復号システム (video encoding and decoding system) 10を示すブロック図である。図1に示されているように、システム10は、宛先デバイス14によって後で復号されるべき符号化ビデオデータ (encoded video data) を与えるソースデバイス12を含む。特に、ソースデバイス12は、コンピュータ可読媒体16を介してビデオデータを宛先デバイス14に与える。ソースデバイス12および宛先デバイス14は、デスクトップコンピュータ、ノートブック (すなわち、ラップトップ) コンピュータ、タブレットコンピュータ、セットトップボックス、いわゆる「スマート」フォンなどの電話ハンドセット、いわゆる「スマート」パッド、テレビジョン、カメラ、ディスプレイデバイス、デジタルメディアプレーヤ、ビデオゲームコンソール、ビデオストリーミングデバイスなどを含む、広範囲にわたるデバイスのいずれかを備え得る。いくつかの場合には、ソースデバイス12および宛先デバイス14は、ワイヤレス通信のために装備され得る。

【0017】

[0028] 宛先デバイス14は、コンピュータ可読媒体16を介して復号されるべき符号化ビデオデータを受信し得る。コンピュータ可読媒体16は、ソースデバイス12から宛先デバイス14に符号化ビデオデータを移動させることができ任意のタイプ (type) の媒体またはデバイスを備え得る。一例では、コンピュータ可読媒体16は、ソースデバイス12が、符号化ビデオデータを宛先デバイス14にリアルタイムで直接送信することを可能にするための通信媒体を備え得る。符号化ビデオデータは、ワイヤレス通信プロトコルなどの通信規格に従って変調され、宛先デバイス14に送信され得る。通信媒体は、無線周波数 (RF) スペクトルまたは1つまたは複数の物理伝送線路など、任意のワイヤレスまたはワイヤード通信媒体を備え得る。通信媒体は、ローカルエリアネットワーク、ワイドエリアネットワーク、またはインターネットなどのグローバルネットワークなど、パケットベースネットワークの一部を形成し得る。通信媒体は、ソースデバイス12から宛先デバイス14への通信を可能にするために有用であり得るルータ、スイッチ、基地局、または任意の他の機器を含み得る。

【0018】

[0029] いくつかの例では、符号化データは、出力インターフェース22からストレージデバイスに出力され得る。同様に、符号化データは、入力インターフェースによってス

10

20

30

40

50

トレージデバイスからアクセスされ得る。ストレージデバイスは、ハードドライブ、B1u-ray (登録商標) ディスク、DVD、CD-ROM、フラッシュメモリ、揮発性または不揮発性メモリ、あるいは符号化ビデオデータを記憶するための任意の他の好適なデジタル記憶媒体など、様々な分散されたまたはローカルにアクセスされるデータ記憶媒体のいずれかを含み得る。さらなる一例では、ストレージデバイスは、ソースデバイス12によって生成された符号化ビデオを記憶し得るファイルサーバまたは別の中間ストレージデバイスに対応し得る。宛先デバイス14は、ストリーミングまたはダウンロードを介して、ストレージデバイスから記憶されたビデオデータにアクセスし得る。ファイルサーバは、符号化ビデオデータを記憶することと、その符号化ビデオデータを宛先デバイス14に送信することとが可能な任意のタイプのサーバであり得る。例示的なファイルサーバとしては、(たとえば、ウェブサイトのための) ウェブサーバ、FTPサーバ、ネットワーク接続ストレージ (NAS : network attached storage) デバイス、またはローカルディスクドライブがある。宛先デバイス14は、インターネット接続を含む、任意の標準のデータ接続を通して符号化ビデオデータにアクセスし得る。これは、ファイルサーバに記憶された符号化ビデオデータにアクセスするのに好適であるワイヤレスチャネル (たとえば、Wi-Fi (登録商標) 接続)、ワイヤード接続 (たとえば、DSL、ケーブルモ뎀など)、またはその両方の組合せを含み得る。ストレージデバイスからの符号化ビデオデータの送信は、ストリーミング送信、ダウンロード送信、またはそれらの組合せであり得る。

【0019】

[0030] 本開示の技法は、必ずしもワイヤレス適用例または設定に限定されるとは限らない。本技法は、オーバージエアテレビジョン放送、ケーブルテレビジョン送信、衛星テレビジョン送信、動的適応ストリーミングオーバーHTTP (DASH : dynamic adaptive streaming over HTTP)などのインターネットストリーミングビデオ送信、データ記憶媒体上に符号化されたデジタルビデオ、データ記憶媒体に記憶されたデジタルビデオの復号、または他の適用例など、様々なマルチメディア適用例のいずれかをサポートするビデオコーディングに適用され得る。いくつかの例では、システム10は、ビデオストリーミング、ビデオ再生、ビデオブロードキャスティング、および/またはビデオテレフォニーなどの適用例をサポートするために、一方向または双方向のビデオ送信をサポートするように構成され得る。

【0020】

[0031] 図1の例では、ソースデバイス12は、ビデオソース18と、ビデオエンコーダ20と、出力インターフェース22とを含む。宛先デバイス14は、入力インターフェース28と、ビデオデコーダ30と、ディスプレイデバイス31とを含む。本開示によれば、ソースデバイス12のビデオエンコーダ20は、拡張CABAC設計に従ってデータをコーディングするための技法を適用するように構成され得る。他の例では、ソースデバイスおよび宛先デバイスは他の構成要素または構成を含み得る。たとえば、ソースデバイス12は、外部カメラなど、外部ビデオソース18からビデオデータを受信し得る。同様に、宛先デバイス14は、内蔵ディスプレイデバイスを含むのではなく、外部ディスプレイデバイスとインターフェースし得る。

【0021】

[0032] 図1の図示のシステム10は一例にすぎない。拡張CABAC設計に従ってデータをコーディングするための技法は、任意のデジタルビデオ符号化および/または復号デバイスによって実行され得る。概して、本開示の技法はビデオ符号化デバイスによって実行されるが、本技法は、一般に「コーデック (CODEC)」と呼ばれるビデオエンコーダ/デコーダによっても実行され得る。その上、本開示の技法はビデオプリプロセッサによつても実行され得る。ソースデバイス12および宛先デバイス14は、ソースデバイス12が宛先デバイス14に送信するためのコード化ビデオデータを生成するような、コーディングデバイスの例にすぎない。いくつかの例では、デバイス12、14は、デバイス12、14の各々がビデオ符号化構成要素とビデオ復号構成要素 (video encoding and dec

10

20

30

40

50

oding components) とを含むように、実質的に対称的に動作し得る。したがって、システム 10 は、たとえば、ビデオストリーミング、ビデオ再生、ビデオプロードキャスティング、またはビデオテレフォニーのための、ビデオデバイス 12 とビデオデバイス 14 との間の一方向または双方向のビデオ送信をサポートし得る。

【 0 0 2 2 】

[0033] ソースデバイス 12 のビデオソース 18 は、ビデオカメラなどのビデオキャプチャデバイス、以前にキャプチャ (capture) されたビデオを含んでいるビデオアーカイブ、および / またはビデオコンテンツプロバイダからビデオを受信するためのビデオフィードインターフェースを含み得る。さらなる代替として、ビデオソース 18 は、ソースビデオとしてのコンピュータグラフィックスベースデータ、またはライブビデオとアーカイブビデオとコンピュータ生成ビデオとの組合せを生成し得る。いくつかの場合には、ビデオソース 18 がビデオカメラである場合、ソースデバイス 12 および宛先デバイス 14 は、いわゆるカメラフォンまたはビデオフォンを形成し得る。ただし、上述のように、本開示で説明される技法は、概してビデオコーディングに適用可能であり得、ワイヤレスおよび / またはワイヤード適用例に適用され得る。各場合において、キャプチャされたビデオ、前にキャプチャされたビデオ、またはコンピュータ生成ビデオは、ビデオエンコーダ 20 によって符号化され得る。符号化ビデオ情報は、次いで、出力インターフェース 22 によってコンピュータ可読媒体 16 上に出力され得る。

【 0 0 2 3 】

[0034] コンピュータ可読媒体 (Computer-readable medium) 16 は、ワイヤレスプロードキャストまたはワイヤードネットワーク送信などの一時媒体、あるいはハードディスク、フラッシュドライブ、コンパクトディスク、デジタルビデオディスク、B l u - r a y ディスク、または他のコンピュータ可読媒体などの記憶媒体 (すなわち、非一時的記憶媒体 (non-transitory storage media)) を含み得る。いくつかの例では、ネットワークサーバ (図示せず) は、たとえば、ネットワーク送信を介して、ソースデバイス 12 から符号化ビデオデータを受信し、その符号化ビデオデータを宛先デバイス 14 に与え得る。同様に、ディスクスタンピング設備など、媒体製造設備のコンピューティングデバイスは、ソースデバイス 12 から符号化ビデオデータを受信し、その符号化ビデオデータを含んでいるディスクを生成し得る。ビデオ復号デバイスによって処理されるとき、ディスク上の符号化ビデオデータは、ビデオ復号デバイスに、本明細書で開示される様々な例に従ってビデオデータを復号させ得る。したがって、コンピュータ可読媒体 16 は、様々な例において、様々な形態の 1 つまたは複数のコンピュータ可読媒体を含むことが理解されよう。

【 0 0 2 4 】

[0035] 宛先デバイス 14 の入力インターフェース 28 は、コンピュータ可読媒体 16 から情報を受信する。コンピュータ可読媒体 16 の情報は、ビデオエンコーダ 20 によって定義され、またビデオデコーダ 30 によって使用される、ブロックおよび他のコード化ユニット、たとえば、G O P の特性および / または処理を記述するシンタックス要素を含む、シンタックス情報を含み得る。ディスプレイデバイス 31 は、復号ビデオデータ (decoded video data) をユーザに対して表示し、陰極線管 (C R T)、液晶ディスプレイ (L C D)、プラズマディスプレイ、有機発光ダイオード (O L E D) ディスプレイ、または別のタイプのディスプレイデバイスなど、様々なディスプレイデバイスのいずれかを備え得る。

【 0 0 2 5 】

[0036] ビデオエンコーダ 20 およびビデオデコーダ 30 は、I T U - T H . 2 6 5 とも呼ばれる、高効率ビデオコーディング (H E V C) 規格など、ビデオコーディング規格に従って動作し得る。代替的に、ビデオエンコーダ 20 およびビデオデコーダ 30 は、代替的にM P E G - 4 , P a r t 1 0 , アドバンストビデオコーディング (A V C) と呼ばれるI T U - T H . 2 6 4 規格など、他のプロプライエタリ規格または業界規格 (proprietary or industry standards) 、あるいはそのような規格の拡張に従って動作し得

10

20

30

40

50

る。ただし、本開示の技法は、いかなる特定のコーディング規格にも限定されない。ビデオコーディング規格の他の例としては、MPEG-2 およびITU-T H.263 がある。図1には示されていないが、いくつかの態様では、ビデオエンコーダ20 およびビデオデコーダ30 は、それぞれ、オーディオエンコーダおよびデコーダと統合され得、共通のデータストリームまたは別個のデータストリーム中のオーディオとビデオの両方の符号化を処理するために、適切なMUX-DEMUXユニット、または他のハードウェアおよびソフトウェアを含み得る。適用可能な場合、MUX-DEMUXユニットは、ITU H.223 マルチプレクサプロトコル、またはユーザデータグラムプロトコル(UDP: user datagram protocol)などの他のプロトコルに準拠し得る。

【0026】

10

[0037] ビデオエンコーダ20 およびビデオデコーダ30 はそれぞれ、1つまたは複数のマイクロプロセッサ(microprocessor)、デジタル信号プロセッサ(DSP)、特定用途向け集積回路(ASIC)、フィールドプログラマブルゲートアレイ(FPGA)、ディスクリート論理、ソフトウェア、ハードウェア、ファームウェアなど、様々な好適なエンコーダ回路のいずれか、あるいはそれらの任意の組合せとして実装され得る。本技法が部分的にソフトウェアで実装されるとき、デバイスは、ソフトウェアのための命令を好適な非一時的コンピュータ可読媒体に記憶し、本開示の技法を実行するために1つまたは複数のプロセッサを使用してハードウェアでその命令を実行し得る。ビデオエンコーダ20 およびビデオデコーダ30 の各々は1つまたは複数のエンコーダまたはデコーダ中に含まれ得、そのいずれも、それぞれのデバイスにおいて複合エンコーダ/デコーダ(コーデック)の一部として統合され得る。

【0027】

20

[0038] 概して、HEVCによれば、ビデオフレームまたはピクチャは、ルーマサンプルとクロマサンプルの両方を含む一連のツリーブロックまたは最大コーディングユニット(LCU: largest coding unit)に分割され得る。ビットストリーム(bitstream)内のシンタックスデータが、ピクセルの数に関して最大コーディングユニットであるLCUのサイズを定義し得る。スライスは、コーディング順序でいくつかの連続するツリーブロックを含む。ビデオフレームまたはピクチャは、1つまたは複数のスライスに区分され得る。各ツリーブロックは、4分木に従ってコーディングユニット(CU: coding unit)にスプリットされ得る。概して、4分木データ構造はCUごとに1つのノードを含み、ルートノードはツリーブロックに対応する。CUが4つのサブCUにスプリットされた場合、CUに対応するノードは4つのリーフノード(leaf node)を含み、リーフノードの各々はサブCUのうちの1つに対応する。

30

【0028】

[0039] 4分木データ構造の各ノードは、対応するCUのためのシンタックスデータを与え得る。たとえば、4分木中のノードは、そのノードに対応するCUがサブCUにスプリットされるかどうかを示すスプリットフラグ(split flag)を含み得る。CUのためのシンタックス要素は、再帰的に定義され得、CUがサブCUにスプリットされるかどうかに依存し得る。CUがさらにスプリットされない場合、そのCUはリーフCUと呼ばれる。本開示では、元のリーフCUの明示的スプリッティングが存在しない場合でも、リーフCUの4つのサブCUはリーフCUとも呼ばれる。たとえば、 16×16 サイズのCUがさらにスプリットされない場合、その 16×16 CUが決してスプリットされなくても、4つの 8×8 サブCUはリーフCUとも呼ばれる。

40

【0029】

[0040] CUは、CUがサイズ差異を有しないことを除いて、H.264規格のマクロブロックと同様の目的を有する。たとえば、ツリーブロックは、(サブCUとも呼ばれる)4つの子ノードにスプリットされ得、各子ノードは、今度は親ノードとなり、別の4つの子ノードにスプリットされ得る。4分木のリーフノードと呼ばれる、最後のスプリットされていない子ノードは、リーフCUとも呼ばれるコーディングノードを備える。コード化ビットストリームに関連するシンタックスデータは、最大CU深度(maximum CU depth

50

)と呼ばれる、ツリープロックがスプリットされ得る最大回数を定義し得、また、コーディングノードの最小サイズを定義し得る。それに応じて、ビットストリームは最小コーディングユニット (SCU : smallest coding unit) をも定義し得る。本開示は、HEVCのコンテキストにおけるCU、予測ユニット (PU : prediction unit)、または変換ユニット (TU : transform unit)、あるいは他の規格のコンテキストにおける同様のデータ構造 (たとえば、H.264 / AVCにおけるマクロブロックおよびそのサブブロック) のいずれかを指すために「ブロック (block)」という用語を使用する。

【0030】

[0041] CUは、コーディングノードと、コーディングノードに関連する予測ユニット (PU) および変換ユニット (TU) とを含む。CUのサイズは、コーディングノードのサイズに対応し、概して形状が正方形である。CUのサイズは、 8×8 ピクセルから最大サイズ、たとえば、 64×64 以上のピクセルをもつツリープロックのサイズまでに及び得る。各CUは、1つまたは複数のPUと、1つまたは複数のTUとを含んでいることがある。CUに関連するシンタックスデータは、たとえば、1つまたは複数のPUへのCUの区分を記述し得る。区分モードは、CUが、スキップモード符号化またはダイレクトモード符号化されるか、イントラ予測モード符号化されるか、あるいはインター予測モード符号化されるかの間で異なり得る。PUは、形状が非正方形になるように区分され得る。CUに関連するシンタックスデータは、たとえば、4分木に従う1つまたは複数のTUへのCUの区分をも記述し得る。TUは、形状が正方形または非正方形 (たとえば、矩形) であり得る。

10

【0031】

[0042] HEVC規格は、CUごとに異なり得るTUに従う変換を可能にする。TUは、一般に、区分されたLCUについて定義された所与のCU内のPUのサイズに基づいてサイズ決定されるが、これは常にそうであるとは限らない。TUは、一般に、PUと同じサイズであるかまたはPUよりも小さい。いくつかの例では、CUに対応する残差サンプルは、「残差4分木」 (RQT : residual quad tree) として知られる4分木構造を使用して、より小さいユニットに再分割され得る。RQTのリーフノードは変換ユニット (TU) と呼ばれることがある。TUに関連するピクセル差分値は、変換係数を生成するために変換され得、その変換係数は量子化され得る。

20

【0032】

[0043] リーフCUは1つまたは複数の予測ユニット (PU) を含み得る。概して、PUは、対応するCUの全部または一部分に対応する空間エリアを表し、そのPUの参照サンプルを取り出しえる / または生成するためのデータを含み得る。その上、PUは、予測に関係するデータを含む。たとえば、PUがイントラモード符号化されるとき、PUのためのデータは、PUに対応するTUのためのイントラ予測モードを記述するデータを含み得る残差4分木 (RQT) 中に含まれ得る。RQTは変換ツリーと呼ばれることもある。いくつかの例では、イントラ予測モードは、RQTの代わりに、リーフCUシンタックス中でシグナリングされ得る。別の例として、PUがインターモード符号化されるとき、PUは、PUのための、1つまたは複数の動きベクトルなど、動き情報を定義するデータを含み得る。PUのための動きベクトルを定義するデータは、たとえば、動きベクトルの水平成分、動きベクトルの垂直成分、動きベクトルについての解像度 (たとえば、 $1/4$ ピクセル精度または $1/8$ ピクセル精度)、動きベクトルが指す参照ピクチャ、および / または動きベクトルのための参照ピクチャリスト (たとえば、リスト0、リスト1、またはリストC) を記述し得る。

30

【0033】

[0044] 1つまたは複数のPUを有するリーフCUはまた、1つまたは複数の変換ユニット (TU) を含み得る。変換ユニットは、上記で説明されたように、(TU4分木構造とも呼ばれる) RQTを使用して指定され得る。たとえば、スプリットフラグは、リーフCUが4つの変換ユニットにスプリットされるかどうかを示し得る。次いで、各変換ユニットは、さらなるサブTUにさらにスプリットされ得る。TUがさらにスプリットされな

40

50

いとき、その T U はリーフ T U と呼ばれることがある。概して、イントラコーディングの場合、リーフ C U に属するすべてのリーフ T U は同じイントラ予測モードを共有する。すなわち、概して、リーフ C U のすべての T U の予測値を計算するために同じイントラ予測モードが適用される。イントラコーディングでは、ビデオエンコーダは、イントラ予測モードを使用して各リーフ T U の残差値を、 T U に対応する C U の一部と元のブロックとの間の差分として計算し得る。 T U は、必ずしも P U のサイズに制限されるとは限らない。したがって、 T U は、 P U よりも大きいことも小さいこともある。イントラコーディングでは、 P U は、同じ C U のための対応するリーフ T U とコロケート (collocate) され得る。いくつかの例では、リーフ T U の最大サイズは、対応するリーフ C U のサイズに対応し得る。

10

【 0 0 3 4 】

[0045] その上、リーフ C U の T U はまた、残差 4 分木 (RQT) と呼ばれる、それぞれの 4 分木データ構造に関連し得る。すなわち、リーフ C U は、リーフ C U がどのように T U に区分されるかを示す 4 分木を含み得る。 T U 4 分木のルートノードは概してリーフ C U に対応し、 C U 4 分木のルートノードは概してツリーブロック (またはLCU) に対応する。スプリットされない RQT の T U はリーフ T U と呼ばれる。概して、本開示では、別段に記載されていない限り、リーフ C U およびリーフ T U に言及するためにそれぞれ C U および T U という用語を使用する。

【 0 0 3 5 】

[0046] ビデオシーケンスは、一般に、一連のビデオフレームまたはピクチャを含む。ピクチャグループ (GOP : group of pictures) は、概して、ビデオピクチャのうちの一連の 1 つまたは複数を備える。 GOP は、 GOP 中に含まれるいくつかのピクチャを記述するシンタックスデータを、 GOP のヘッダ中、ピクチャのうちの 1 つまたは複数のヘッダ中、または他の場所に含み得る。ピクチャの各スライスは、それぞれのスライスのための符号化モードを記述するスライスシンタックスデータを含み得る。ビデオエンコーダ 20 は、一般に、ビデオデータを符号化するために個々のビデオスライス内のビデオブロックに対して動作する。ビデオブロックは C U 内のコーディングノードに対応し得る。ビデオブロックは、固定サイズまたは可変サイズを有し得、指定されたコーディング規格に応じてサイズが異なり得る。

20

【 0 0 3 6 】

[0047] 一例として、予測は様々なサイズの P U について実行され得る。特定の C U のサイズが $2N \times 2N$ であると仮定すると、イントラ予測が、 $2N \times 2N$ または $N \times N$ の P U サイズに対して実行され得、インター予測が、 $2N \times 2N$ 、 $2N \times N$ 、 $N \times 2N$ 、または $N \times N$ の対称的な P U サイズに対して実行され得る。インター予測のための非対称区分は、 $2N \times nU$ 、 $2N \times nD$ 、 $nL \times 2N$ 、および $nR \times 2N$ の P U サイズについても実行され得る。非対称区分では、 C U の一方向は区分されないが、他の方向は 25 % と 75 % とに区分される。25 % の区分に対応する C U の部分は、「n」とその後ろに付く「U p」、「D o w n」、「L e f t」、または「R i g h t」という表示によって示される。したがって、たとえば、「 $2N \times nU$ 」は、上部の $2N \times 0.5N$ P U と下部の $2N \times 1.5N$ P U とで水平方向に区分された $2N \times 2N$ C U を指す。

30

【 0 0 3 7 】

[0048] 本開示では、「 $N \times N$ (NxN)」および「 $N \times N$ (N by N)」は、垂直寸法および水平寸法に関するビデオブロックのピクセル寸法、たとえば、 16×16 (16x16) ピクセルまたは 16×16 (16 by 16) ピクセルを指すために互換的に使用され得る。概して、 16×16 ブロックは、垂直方向に 16 ピクセルを有し ($y = 16$) 、水平方向に 16 ピクセルを有する ($x = 16$)。同様に、 $N \times N$ ブロックは、概して、垂直方向に N ピクセルを有し、水平方向に N ピクセルを有し、ここで、N は非負整数値を表す。ブロック中のピクセルは行および列に配列され得る。その上、ブロックは、必ずしも、水平方向において垂直方向と同じ数のピクセルを有する必要があるとは限らない。たとえば、ブロックは $N \times M$ ピクセルを備え得、ここで、M は必ずしも N に等しいとは限らない。

40

50

【0038】

[0049] C U の P U を使用したイントラ予測コーディングまたはインター予測コーディングの後に、ビデオエンコーダ 20 は、C U の T U のための残差データを計算し得る。P U は、(ピクセル領域とも呼ばれる) 空間領域において予測ピクセルデータを生成する方法またはモードを記述するシンタックスデータを備え得、T U は、変換、たとえば、残差ビデオデータへの離散コサイン変換 (DCT)、整数変換、ウェーブレット変換、または概念的に同様の変換の適用後に、変換領域において係数を備え得る。残差データは、符号化されていないピクチャのピクセルと、P U に対応する予測値との間のピクセル差分に対応し得る。ビデオエンコーダ 20 は、C U のための残差データを表す量子化された変換係数を含むように T U を形成し得る。すなわち、ビデオエンコーダ 20 は、(残差ブロックの形態の) 残差データを計算し、変換係数のブロックを生成するために残差ブロックを変換し、次いで、量子化された変換係数を形成するために変換係数を量子化し得る。ビデオエンコーダ 20 は、量子化された変換係数を含む T U、ならびに他のシンタックス情報(たとえば、T U のためのスプリッティング情報)を形成し得る。10

【0039】

[0050] 上述のように、変換係数を生成するための任意の変換の後に、ビデオエンコーダ 20 は、変換係数の量子化を実行し得る。量子化は、概して、係数を表すために使用されるデータの量をできるだけ低減するために変換係数が量子化され、さらなる圧縮を行うプロセスを指す。量子化プロセスは、係数の一部または全部に関連するビット深度 (bit depth) を低減し得る。たとえば、量子化中に n ビット値が m ビット値に切り捨てられ得、ここで、n は m よりも大きい。20

【0040】

[0051] 量子化の後に、ビデオエンコーダは、変換係数を走査し、量子化された変換係数を含む 2 次元行列から 1 次元ベクトルを生成し得る。走査は、アレイの前部により高いエネルギー(したがって、より低い周波数)係数を配置し、アレイの後部により低いエネルギー(したがって、より高い周波数)係数を配置するように設計され得る。いくつかの例では、ビデオエンコーダ 20 は、エントロピー符号化 (entropy encode) され得るシリアル化ベクトルを生成するために、量子化された変換係数を走査するためにあらかじめ定義された走査順序を利用し得る。他の例では、ビデオエンコーダ 20 は適応型走査を実行し得る。1 次元ベクトルを形成するために、量子化された変換係数を走査した後に、ビデオエンコーダ 20 は、たとえば、本開示で説明されるコンテキスト適応型バイナリ算術コーディング (CABAC) 設計に従って、1 次元ベクトルをエントロピー符号化し得る。ビデオエンコーダ 20 はまた、ビデオデータを復号する際にビデオデコーダ 30 が使用するための符号化ビデオデータに関連するシンタックス要素をエントロピー符号化し得る。30

【0041】

[0052] 概して、ビデオデコーダ 30 は、符号化データを復号するためにビデオエンコーダ 20 によって実行されるものと、相反するが、実質的に同様のプロセスを実行する。たとえば、ビデオデコーダ 30 は、残差ブロックを再生するために、受信された T U の係数を逆量子化および逆変換する。ビデオデコーダ 30 は、予測されたブロックを形成するために、シグナリングされた予測モード(イントラ予測またはインター予測)を使用する。次いで、ビデオデコーダ 30 は、元のブロックを再生するために、(ピクセルごとに) 予測されたブロックと残差ブロックとを組み合わせる。ブロック境界に沿って視覚的アーティファクト (visual artifact) を低減するためにデブロッキングプロセスを実行することなど、追加の処理が実行され得る。さらに、ビデオデコーダ 30 は、ビデオエンコーダ 20 の CABAC 符号化プロセスに相反するが、それと実質的に同様の様式で CABAC を使用してシンタックス要素を復号し得る。40

【0042】

[0053] 本開示は、概して、ビデオエンコーダ 20 が、ある情報をビデオデコーダ 30 などの別のデバイスに「シグナリング (signaling)」することに言及することがある。ただし、ビデオエンコーダ 20 は、いくつかのシンタックス要素をビデオデータの様々50

符号化部分に関連付けることによって情報をシグナリングし得ることを理解されたい。すなわち、ビデオエンコーダ20は、いくつかのシンタックス要素をビデオデータの様々な符号化部分のヘッダに記憶することによってデータを「シグナリング」し得る。いくつかの場合には、そのようなシンタックス要素は、ビデオデコーダ30によって受信され、復号されるより前に、符号化され、記憶され（たとえば、ストレージデバイス32に記憶され）得る。したがって、「シグナリング」という用語は、通信がリアルタイムまたはほぼリアルタイムで行われるか、あるいは、符号化時にシンタックス要素を媒体に記憶し、次いで、この媒体に記憶された後の任意の時間にそのシンタックス要素が復号デバイスによって取り出され得るときなどに行われ得る、ある時間期間にわたって行われるかどうかにかかわらず、概して、圧縮ビデオデータを復号するためのシンタックスまたは他のデータの通信を指し得る。

10

【0043】

[0054] 以下のセクションは、BACおよびCABAC技法についてより詳細に説明する。BACは、概して、再帰的間隔再分割プロセッサ（recursive interval-subdividing procedure）である。BACは、H.264/AVCおよびH.265/HEVCビデオコーディング規格におけるCABACプロセスにおいてビンを符号化するために使用される。BACコーダの出力は、最終コード化確率間隔（final coded probability interval）内の確率に対する値またはポインタを表すバイナリストリームである。確率間隔（probability interval）は、範囲（range）および下端値（lower end value）によって指定される。範囲は確率間隔の拡張である。低はコーディング間隔の下限である。

20

【0044】

[0055] ビデオコーディングへの算術コーディングの適用は、D. Marpe、H. S. Schwarz、およびT. Wiegand「Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard」、IEEE Transactions on Circuits and Systems for Video Technology、vol. 13、no. 7、2003年7月に記載されている。CABACは、3つの主要な機能、すなわち、2値化（binarization）、コンテキストモデリング（context modeling）、および算術コーディング（arithmetic coding）を伴う。2値化は、シンタックス要素をバイナリシンボル（または「ビン（bin）」）にマッピング（mapping）する機能を指す。バイナリシンボルは「ビンストリング（bin string）」と呼ばれることがある。コンテキストモデリングは、様々なビンの確率を推定する機能を指す。算術コーディングは、推定された確率に基づいて、ビンをビットに圧縮する後続の機能を指す。バイナリ算術コーダなど、様々なデバイスおよび/またはそれらのモジュールは算術コーディングの機能を実行し得る。

30

【0045】

[0056] HEVCでは、単項（U）、短縮単項（TU：truncated unary）、k次指数ゴロム（EGk：kth-order Exp-Golomb）、および固定長（FL：fixed length）を含む、いくつかの異なる2値化プロセスが使用される。様々な2値化プロセスの詳細は、V. SzeおよびM. Budagavi、「High throughput CABAC entropy coding in HEVC」、IEEE Transactions on Circuits and Systems for Video Technology（TCST），vol. 22、no. 12、1778～1791ページ、2012年12月に記載されている。

40

【0046】

[0057] CABACにおける各コンテキスト（すなわち、確率モデル）は状態によって表される。各状態（）は、特定のシンボル（たとえば、ビン）が劣勢シンボル（LPS：Least Probable Symbol）である確率（p）を暗黙的に表す。シンボルはLPSまたは優勢シンボル（MPS：Most Probable Symbol）であり得る。シンボルはバイナリであり、したがって、MPSおよびLPSは0または1であり得る。確率は、対応するコンテキストについて推定され、算術コーダ（arithmetic coder）を使用してシンボルをエンコードピーコーディングするために（暗黙的に）使用される。

50

【0047】

[0058] B A C のプロセスは、コーディングすべきコンテキストとコーディングされているビンの値とに応じて、その内部値「範囲」および「低」を変更する状態機械によって扱われる。コンテキストの状態（すなわち、その確率）に応じて、範囲は、範囲 M P S（状態における優勢シンボルの範囲）と範囲 L P S（状態における劣勢シンボルの範囲）とに分割される。理論上、確率状態の範囲 L P S 値は以下の乗算によって導出される。

【0048】

【数1】

$$\text{範囲LPS } \sigma = \text{範囲} \times p_\sigma$$

10

【0049】

上式で、 p_σ は、L P S を選択する確率である。もちろん、M P S の確率は $1 - p_\sigma$ である。等価的に、範囲 M P S は、範囲 - 範囲 L P S に等しい。B A C は、コーディングすべきコンテキストビンの状態と、現在の範囲と、コーディングされているビンの値（すなわち、ビンが L P S に等しいのか M P S に等しいのか）とに応じて、範囲を反復的に更新する。

【0050】

[0059] 図 2 A および図 2 B は、ビン n におけるこのプロセスの例を示す。図 2 A の例 100 では、ビン n において、ビン 2 における範囲（range）は、あるコンテキスト状態（）を仮定すると、L P S (p_σ) の確率によって与えられる範囲 M P S と範囲 L P S とを含む。例 100 は、ビン n の値が M P S に等しいときのビン $n + 1$ における範囲の更新を示す。この例では、低（low）は同じままであるが、ビン $n + 1$ における範囲の値は、ビン n における範囲 M P S の値に低減される。図 2 B の例 102 は、ビン n の値が M P S に等しくない（すなわち、L P S に等しい）ときのビン $n + 1$ における範囲の更新を示す。この例では、低は、ビン n における範囲 L P S の下側範囲値に移動される。さらに、ビン $n + 1$ における範囲の値は、ビン n における範囲 L P S の値に低減される。

20

【0051】

[0060] H E V C では、範囲は 9 ビットで表され、低は 10 ビットで表される。範囲値および低値を十分な精度で維持するための再正規化プロセス（renormalization process）がある。範囲が 256 よりも小さいときはいつでも、再正規化が行われる。したがって、範囲は、再正規化の後、常に 256 に等しいかまたはそれよりも大きい。範囲の値と低の値とに応じて、B A C は、ビットストリームに「0」または「1」を出力するか、または将来の出力のために保持するために（B O：未解決ビット（bits-outstanding）と呼ばれる）内部変数を更新する。図 3 は、範囲に応じた B A C 出力の例を示す。たとえば、範囲および低が、あるしきい値（たとえば、512）を上回るとき、ビットストリームに「1」が出力される。範囲および低が、あるしきい値（たとえば、512）を下回るとき、ビットストリームに「0」が出力される。範囲および下側が、あるしきい値間にあるとき、ビットストリームに何も出力されない。代わりに、B O 値が増分され、次のビンが符号化される。

30

【0052】

[0061] H E V C の C A B A C コンテキストモデルでは、128 個の状態がある。0 から 63 まであり得る（状態によって示される）64 個の可能な L P S 確率がある。各 M P S は 0 または 1 であり得る。したがって、128 個の状態は、64 個の状態確率 \times M P S のための 2 個の可能な値（0 または 1）である。したがって、確率モデルは 7 ビットエントリ（entry）として記憶され得る。各 7 ビットエントリにおいて、確率状態を表すために 6 ビットが割り振られ得、適用可能なコンテキストメモリ中の優勢シンボル（M P S）のために 1 ビットが割り振られ得る。

40

【0053】

[0062] L P S 範囲（範囲 L P S）を導出する計算を低減するために、H E V C では、すべての場合についての結果が事前計算され、近似値としてルックアップテーブル（lo

50

ok-up table) に記憶される。したがって、LPS 範囲は、単純なテーブルルックアップを使用することによって、乗算なしに取得され得る。乗算は、多くのハードウェアアーキテクチャにおいて有意なレイテンシ (latency) を生じ得るので、この動作を回避することは、いくつかのデバイスまたはアプリケーションにとって重要であり得る。

【0054】

[0063] 4 列事前計算 LPS 範囲テーブルが乗算の代わりに使用され得る。範囲は 4 つのセグメントに分割される。セグメントインデックスは、質問 (範囲 $>> 6$) & 3 によって導出され得る。事実上、セグメントインデックスは、実際の範囲からビットをシフトし、ドロップすることによって導出される。以下の表 1 は、可能な範囲およびそれらの対応するインデックスを示す。

【0055】

【表 1】

表1-範囲インデックス

範囲	256-319	320-383	384-447	448-511
(範囲 $>> 6$) & 3	0	1	2	3

【0056】

[0064] LPS 範囲テーブルは、次いで、64 個のエントリ (確率状態ごとに 1 つ) \times 4 (範囲インデックスごとに 1 つ) を有する。各エントリは、範囲 LPS、すなわち、範囲に LPS 確率を乗算した値である。このテーブルの部分の一例が以下の表 2 に示される。表 2 は、確率状態 9 ~ 12 を示す。HEVC のための 1 つの提案では、確率状態は 0 ~ 63 にわたり得る。

【0057】

【表 2】

表2-範囲LPS

確率状態(σ)	範囲LPS			
	インデックス0	インデックス1	インデックス2	インデックス3
...
9	90	110	130	150
10	85	104	123	142
11	81	99	117	135
12	77	94	111	128
...

【0058】

[0065] 各セグメント (すなわち、範囲値) において、各確率状態 の LPS 範囲があらかじめ定義される。言い換えれば、確率状態 の LPS 範囲が 4 つの値 (すなわち、範囲インデックスごとに 1 つの値) に量子化される。所与のポイントにおいて使用される特定の LPS 範囲は、範囲がどのセグメントに属するかに依存する。テーブル内で使用される可能な LPS 範囲の数は、テーブル列の数 (すなわち、可能な LPS 範囲値の数) と LPS 範囲精度との間のトレードオフである。概して、より多数の列は、LPS 範囲値のより小さい量子化誤差を生じるが、また、テーブルを記憶するためにより多くのメモリの必要を増加させる。より少数の列は、量子化誤差を増加させるが、また、テーブルを記憶するために必要とされるメモリを低減する。

【0059】

[0066] 上記で説明されたように、各 LPS 確率状態は、対応する確率を有する。HE

10

20

30

40

50

V C では、 64 個の代表的確率値 p [0 . 0 1 8 7 5 , 0 . 5] が、 再帰的式である以下の式 (1) に従って L P S (劣勢シンボル) について導出される。

【 0 0 6 0 】

【 数 2 】

$$p_{\sigma} = \alpha * p_{\sigma-1} \text{ すべての } \sigma = 1, \dots, 63 \text{ である場合}$$

ここで $\alpha = \left(\frac{0.01875}{0.5}\right)^{1/63}$

【 0 0 6 1 】

[0067] 上記の例では、 選定されたスケーリングファクタ (scaling factor) 0 . 9 4 9 2 と確率のセットのカーディナリティ (cardinality) $N = 64$ の両方が、 確率表現の精度と高速適応のための要望との間の良好な妥協を表す。いくつかの例では、 1 により近い の値は、 より高い精度をもつ遅い適応 (「定常状態挙動 (steady-state behavior) 」) を生じ得、 より速い適応は、 精度の低減という犠牲を払って の値を減少させる、 非定常の場合に達成され得る。スケーリングファクタ は、 現在の更新に有意な影響を有する、 前に符号化されたビンの数を示すウィンドウサイズに対応し得る。M P S (優勢シンボル) の確率は、 1 - L P S (劣勢シンボル) の確率に等しい。言い換えれば、 M P S の確率は、 式 (1 - L P S) によって表され得、 ここで、 「 L P S 」 は L P S の確率を表す。したがって、 H E V C における C A B A C によって表され得る確率範囲は、 [0 . 0 1 8 7 5 , 0 . 9 8 1 2 5 (= 1 - 0 . 0 1 8 7 5)] である。

【 0 0 6 2 】

[0068] シンタックス要素のための値のビット (または「ビン (bin) 」) をコーディングするために使用されるコンテキストの確率状態が、 信号統計値 (すなわち、 たとえばシンタックス要素のために、 前にコーディングされたビンの値) に従うために更新されるので、 C A B A C は適応型である。更新プロセスは以下の通りである。所与の確率状態について、 更新は、 状態インデックスと、 L P S または M P S のいずれかとして識別される符号化シンボルの値とに依存する。更新プロセスの結果として、 潜在的に修正された L P S 確率推定値と、 必要な場合、 修正された M P S 値とを含む、 新しい確率状態が導出される。

【 0 0 6 3 】

[0069] 各ビンのコーディングの後にコンテキスト切替えが行われ得る。M P S に等しいビン値の場合、 所与の状態インデックスは単に 1 だけ増分される。これは、 L P S 確率がすでにその最小値にある (または、 等価的に、 最大 M P S 確率が達せられる) 、 状態インデックス 62 において M P S が発生したときを除く、 すべての状態について。この場合、 L P S が参照されるか、 または最後のビン値が符号化されるまで、 状態インデックスは固定のままである (最後のビン値の特殊な場合には特殊終了状態が使用される) 。 L P S が発生したとき、 状態インデックスは、 以下の式に示すように、 状態インデックスをある量だけ減分することによって変更される。このルールは、 概して、 以下の例外とともに L P S の各発生に適用される。 L P S が、 同程度の確率がある場合に対応する、 インデックス = 0 をもつ状態において符号化されたと仮定すると、 状態インデックスは固定のままであるが、 M P S 値は、 L P S の値と M P S の値とが交換されるようにトグルされることになる。すべての他の場合では、 たとえどのシンボルが符号化されたとしても、 M P S 値は改変されない。概して、 ビデオコーダは、 所与の L P S 確率 p_{old} と、 その更新されたカウンターパート (counterpart) p_{new} との間の関係を示す、 以下の式 (2) に従って新しい確率状態を導出し得る。

【 0 0 6 4 】

【 数 3 】

$$p_{new} = \begin{cases} \max(\alpha * p_{old}, p_{62}), & \text{MPSが発生した場合} \\ \alpha * p_{old} + (1 - \alpha), & \text{LPSが発生した場合} \end{cases} \quad (2)$$

【 0 0 6 5 】

[0070] 複雑さを低減するために、 ビデオコーダは、 すべての遷移ルールが、 いくつか

10

20

30

40

50

のエントリをそれぞれ有する高々 2 つのテーブルによって実現され得るように、 C A B A C を実装し得る。一例として、すべての遷移ルールは、 7 ビット符号なし整数値の 128 個のエントリをそれぞれ有する高々 2 つのテーブルによって実現され得る（たとえば、以下の表 3 および表 4）。別の例として、すべての遷移ルールは、 6 ビット符号なし整数値の 63 個のエントリをそれぞれ有する高々 2 つのテーブルによって実現され得る（たとえば、 H E V C の表 9 - 41）。状態インデックス i が与えられれば、更新した後に、ビデオコーダは、新しい状態インデックスとして、 MPS 値がコーディングされるときに $\text{TransIdxMPS}[i]$ を定義し、または LPS 値がコーディングされるときに $\text{TransIdxLPS}[i]$ を定義し得る。

【 0066 】

10

【 表 3 】

表3

```
TransIdxMPS[ 128 ] =
{
  2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,
  18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
  34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49,
  50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65,
  66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81,
  82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97,
  98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113,
  114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 124, 125, 126, 127
};
```

【 0067 】

20

【 表 4 】

表4

```
TransIdxLPS[ 128 ] =
{
  1, 0, 0, 1, 2, 3, 4, 5, 4, 5, 8, 9, 8, 9, 10, 11,
  12, 13, 14, 15, 16, 17, 18, 19, 18, 19, 22, 23, 22, 23, 24, 25,
  26, 27, 26, 27, 30, 31, 30, 31, 32, 33, 32, 33, 36, 37, 36, 37,
  38, 39, 38, 39, 42, 43, 42, 43, 44, 45, 44, 45, 46, 47, 48, 49,
  48, 49, 50, 51, 52, 53, 52, 53, 54, 55, 54, 55, 56, 57, 58, 59,
  58, 59, 60, 61, 60, 61, 60, 61, 62, 63, 64, 65, 64, 65, 66, 67,
  66, 67, 66, 67, 68, 69, 68, 69, 70, 71, 70, 71, 70, 71, 72, 73,
  72, 73, 72, 73, 74, 75, 74, 75, 75, 76, 77, 76, 77, 126, 127
};
```

【 0068 】

30

[0071] いくつかの例では、ビデオコーダは、所与の状態インデックス について、 LPS が観測された場合に新しい更新された状態インデックス $\text{TransIdxLPS}[]$ を決定する、 単一のテーブル TransIdxLPS を用いて状態遷移を決定し得る。 MPS 駆動型遷移は、 1 の固定値による状態インデックスの単純な（飽和）増分によって取得され、 更新された状態インデックス $\text{min} (\dots + 1, 62)$ を生じることができる。

40

【 0069 】

[0072] 上記で説明されたように、コンテキストモデリング（ context modeling ）は、より高いコーディング効率を達成するための寄与ファクタである正確な確率推定（ probability estimation ）を与える。したがって、コンテキストモデリングは適応型プロセスである。異なるコンテキストが異なるビンについて使用され得、コンテキストの確率は、前にコーディングされたビンの値に基づいて更新され得る。同様の分布をもつビンは、しばしば同じコンテキストを共有する。各ビンのためのコンテキストは、シンタックス要素の

50

タイプ、シンタックス要素中のビン位置 (b i n I d x) 、ルーマ / クロマ情報、隣接情報などに基づいて選択され得る。

【 0 0 7 0 】

[0073] 所与のスライスをコーディングする前に、1つまたは複数のあらかじめ定義された値に基づいて確率モデルが初期化される。たとえば、q p によって示される入力量子化パラメータと、i n i t V a l によって示されるあらかじめ定義された値とが与えられれば、(状態およびM P S によって示される)確率モデルの7ビットエントリが以下の式(3)に従って導出され得る。

【 0 0 7 1 】

【数4】

```
qp      = Clip3(0, 51, qp);
傾斜    = ( initVal >>4)*5 - 45;
オフセット = ((initVal &15)<<3)-16;
initState= min( max( 1, ((( 傾斜 * qp )>> 4 )+オフセット)), 126 );
MPS    = (initState >= 64 );
状態インデックス = ( (mpState? (initState - 64) : (63 - initState)) <<1 ) + MPS;
```

(3)

【 0 0 7 2 】

[0074] 導出された状態インデックスはM P S 情報を暗黙的に含む。より詳細には、状態インデックスが偶数値であるとき、M P S 値は0に等しい。逆に、状態インデックスが奇数値であるとき、M P S 値は1に等しい。「i n i t V a l 」の値は、8ビット精度での[0, 255]の範囲内にある。あらかじめ定義された値「i n i t V a l 」はスライス依存である。言い換えれば、確率モデルのためのコンテキスト初期化パラメータの3つのセットは、それぞれI、P、およびBスライス中で1つずつ使用される。このようにして、C A B A C を実行するように構成されたビデオ符号化デバイスは、3つの初期化テーブル間でこれらのスライスタイプ(slice type)のために選定することを可能にされ、したがって、異なるコーディングシナリオおよび / または異なるタイプのビデオコンテンツへのより良い適合が達成され得る。

【 0 0 7 3 】

[0075] H E V C によれば、1つのP(またはB)スライスがB(またはP)スライスを用いて初期化されることを可能にするために、別のツールが適用され得る。逆に、そのツールは、1つのBスライスがPスライスを用いて初期化されることを可能にするために適用され得る。関係するシンタックス要素が、(H E V C のセクション7.3.6.1にに対応する)以下の表5で説明され、表5の後に、関係するセマンティクス(semantics)および復号プロセス(decoding process)が以下で説明される。

【 0 0 7 4 】

10

20

30

【表 5】

表5

slice_segment_header() {		記述子
first_slice_segment_in_pic_flag		u(1)
if(nal_unit_type >= BLA_W_LP && nal_unit_type <= RSV_IRAP_VCL23)		
no_output_of_prior_pics_flag		u(1)
slice_pic_parameter_set_id		ue(v)
if(!first_slice_segment_in_pic_flag) {		
if(dependent_slice_segments_enabled_flag)		
dependent_slice_segment_flag		u(1)
slice_segment_address		u(v)
}		
if(!dependent_slice_segment_flag) {		
for(i = 0; i < num_extra_slice_header_bits; i++)		
slice_reserved_flag[i]		u(1)
slice_type		ue(v)
if(output_flag_present_flag)		
pic_output_flag		u(1)
if(separate_colour_plane_flag == 1)		
colour_plane_id		u(2)
if(nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP)		
{		
slice_pic_order_cnt_lsb		u(v)
short_term_ref_pic_set_sps_flag		u(1)
if(!short_term_ref_pic_set_sps_flag)		
short_term_ref_pic_set(num_short_term_ref_pic_sets)		
else if(num_short_term_ref_pic_sets > 1)		
short_term_ref_pic_set_idx		u(v)
if(long_term_ref_pics_present_flag) {		
...		
}		
}		
if(sps_temporal_mvp_enabled_flag)		
slice_temporal_mvp_enabled_flag		u(1)
}		
if(sample_adaptive_offset_enabled_flag) {		
slice_sao_luma_flag		u(1)
slice_sao_chroma_flag		u(1)
}		
if(slice_type == P slice_type == B) {		

10

20

30

40

num_ref_idx_active_override_flag	u(1)
if(num_ref_idx_active_override_flag) {	
num_ref_idx_l0_active_minus1	ue(v)
if(slice_type == B)	
num_ref_idx_l1_active_minus1	ue(v)
}	
if(lists_modification_present_flag && NumPocTotalCurr > 1)	
ref_pic_lists_modification()	
if(slice_type == B)	
mvd_l1_zero_flag	u(1)
if(cabac_init_present_flag)	
cabac_init_flag	u(1)
if(slice_temporal_mvp_enabled_flag) {	
if(slice_type == B)	
collocated_from_l0_flag	u(1)
if((collocated_from_l0_flag && num_ref_idx_l0_active_minus1 > 0) (!collocated_from_l0_flag && num_ref_idx_l1_active_minus1 > 0))	
collocated_ref_idx	ue(v)
}	
if((weighted_pred_flag && slice_type == P) (weighted_bipred_flag && slice_type == B))	
pred_weight_table()	
five_minus_max_num_merge_cand	ue(v)
}	
...	
byte_alignment()	
}	

【0075】

[0076] 表5のシンタックス要素のためのセマンティクスは以下のように定義され得る。

【0076】

[0077] 1に等しい**cabac_init_present_flag**は、PPSを参照するスライスヘッダ(slice header)中に**cabac_init_flag**が存在することを指定する。0に等しい**cabac_init_present_flag**は、PPSを参照するスライスヘッダ中に**cabac_init_flag**が存在しないことを指定する。

【0077】

[0078] **cabac_init_flag**は、以下で説明される復号プロセスにおいて定義されている、コンテキスト変数のための初期化プロセスにおいて使用される初期化テーブルを決定するための方法を指定する。**cabac_init_flag**が存在しないとき、それは0に等しいと推論される。

【0078】

[0079] 記述子:

[0080] **a e (v)** : コンテキスト適応型算術エンコード化シンタックス要素(context-adaptive arithmetic entropy-coded syntax element)。

【0079】

10

20

30

40

50

[0081] **b (8)** : 任意のパターンのビットストリング (bit string) (8 ビット) を有するバイト。

【 0 0 8 0 】

[0082] **f (n)** : 左ビットが先頭の、(左から右に) 書き込まれた n 個のビットを使用する固定パターンビットストリング。

【 0 0 8 1 】

[0083] **s e (v)** : 左ビットが先頭の、符号付き整数 0 次指數ゴロムコード化シンタクス要素 (signed integer 0-th order Exp-Golomb-coded syntax element)。

【 0 0 8 2 】

[0084] **u (n)** : n 個のビットを使用する符号なし整数。シンタクステーブル中で n が「 v 」であるとき、ビット数は、他のシンタクス要素の値に依存する様式で変動する。

【 0 0 8 3 】

[0085] **u e (v)** : 左ビットが先頭の、符号なし整数 0 次指數ゴロムコード化シンタクス要素 (unsigned integer 0-th order Exp-Golomb-coded syntax element)。

【 0 0 8 4 】

[0086] H E V C の表 9 - 4 は、3 つの初期化タイプの各々について、初期化が必要とされるコンテキストインデックス (c t x I d x) を与える。表 9 - 4 は、さらに、初期化のために必要とされる i n i t V a l u e の値を含むテーブル番号 (c t x T a b l e) を含む。 P および B スライスタイプの場合、 i n i t T y p e の導出は、 c a b a c _ i n i t _ f l a g シンタクス要素の値に依存する。ビデオコーダは、 T t h e 可変 i n i t T y p e は、続く以下の擬似コードによって記述される動作を使用してとして導出されるを導出し得る。

【 0 0 8 5 】

【 数 5 】

```
if( slice_type == I )
    initType = 0
else if( slice_type == P )
    initType = cabac_init_flag ? 2 : 1
else
    initType = cabac_init_flag ? 1 : 2
```

【 0 0 8 6 】

[0087] 新しい算術コーダが、 A l s h i n ら、「Multi-parameter probability up-d a t e for CABAC」、文書 : J C T V C - F 2 5 4 、 J C T - V C o f I T U - T S G 1 6 W P 3 および I S O / I E C J T C 1 / S C 2 9 / W G 1 1 、第 6 回会議 : トリノ、イタリア、2 0 1 1 年 7 月 1 4 ~ 2 2 日 (以下「 J C T V C - F 2 5 4 」) 、および A l s h i n ら、「CE1 (subset B): Multi-parameter probability up-date for CABAC」、文書 : J C T V C - G 7 6 4 、 J C T - V C o f I T U - T S G 1 6 W P 3 および I S O / I E C J T C 1 / S C 2 9 / W G 1 1 、第 7 回会議 : ジュネーブ、スイス、2 0 1 1 年 1 1 月 2 1 ~ 3 0 日 (以下「 J C T V C - G 7 6 4 」) に記載されている。 J C T V C - F 2 5 4 および J C T V - G 7 6 4 では、あらゆる確率が 1 から 3 2 7 6 7 までの整数として表した。したがって、すべての計算は 1 6 ビット精度で行われる。 A V C C A B A C において利用される、確率のためのルックアップテーブル (たとえば、上記で説明された T r a n s I d x M P S および T r a n s I d x L P S) および指數メッシュの代わりに、 J C T V C - F 2 5 4 および J C T V - G 7 6 4 において提案されたコーダは、確率更新のために、均一メッシュと、乗算のない式を用いた明示的計算とを利用する。

【 0 0 8 7 】

[0088] 確率 p_i が、(たとえば、以下の式 (4) によって示されるように) (たとえば、 k が 1 5 に等しい) 0 から 2^k までの整数 P_i である確率インデックスによって表され

10

20

30

40

50

ると仮定する。

【0088】

【数6】

$$p_i = P_i / 2^k \quad (4)$$

【0089】

[0089] (たとえば、以下の式(5)によって示されるように) 現代の算術コーデックにおける確率更新のための最も頻繁に使用される以下の式に従うこと。

【0090】

【数7】

10

$$p_{\text{new}} = \alpha y + (1 - \alpha) p_{\text{old}} \quad (5)$$

【0091】

[0090] 式(5)において、現在のシンボルが優勢シンボル(MPS)と一致する場合、yは「0」に等しく、そうでない場合、yは「1」に等しい。この式(すなわち、式(5))は、劣勢シンボル(LPS)の確率のための推定値を与える。上記の説明と同様に、パラメータは、現在の更新に有意な影響を有する、前に符号化されたビンの数を示すウィンドウサイズに対応し得る。

【0092】

[0091] ウィンドウサイズ(W)が2のべき乗($W = 1 / 2^M$ 、Mは正の整数である)であると仮定した場合、式(4)における p_i が入力 p_{old} として与えられれば、更新された確率インデックスは、式(6)において以下で示される中で書き直され得る。

20

【0093】

【数8】

$$P_i = ((2^k) >> M) + P_i - (P_i >> M) \quad (6)$$

【0094】

[0092] JCTVC-F254およびJCTV-G764によって提案された1確率更新モデルでは、Mは、すべてのコンテキストについて固定であり、1つのレジスタのみが、更新された確率を記録するために使用される。一例では、Mは6に等しく設定される。すなわち、ウィンドウサイズは64に等しい。確率更新プロセスは、以下の式(7)によって表され得る。

30

【0095】

【数9】

$$P_{\text{new}} = ((2^k) >> 6) + P_i - (P_i >> 6) \quad (7)$$

【0096】

[0093] JCTVC-F254およびJCTV-G764によって提案された技法の主要なアイデアは、異なるウィンドウサイズでの(ただ1つではなく)いくつかの確率推定を使用し、それらを次のビン確率予測のために加重平均として組み合わせることである。以下の式(8)および(9)は、JCTVC-F254およびJCTV-G764によって提案された技法の一例を示す。各確率 p_i のための式(8)における計算は独立である。

40

【0097】

【数10】

$$p_{i \text{ new}} = W_i y + (1 - W_i) p_{i \text{ old}} \quad (8)$$

【0098】

【数11】

$$p_{\text{new}} = \sum \beta_i p_{i \text{ new}} \quad (9)$$

【0099】

[0094] 各確率 p_i のための式(8)における計算は独立である。

【0100】

[0095] JCTVC-F254 および JCTV-G764 によって提案された方法では、確率推定のための線形結合は、式(10)および(11)に示されているように、 $W_0 = 16$ および $W_1 = 256$ ($W_i = 1 / \beta_i$) 対応する 2 つの被加数からなる。式(10)および(11)において、最後のコーディングビンが「1」である場合、 $Y = 2^{15}$ あり、最後のコーディングビンが「0」である場合、 $Y = 0$ あり、「>> M」は、M ビットのための右算術シフトである。 10

【0101】

【数12】

$$P_0 = (Y >> 4) + P_0 - (P_0 >> 4) \quad (10)$$

【0102】

【数13】

$$P_1 = (Y >> 8) + P_1 - (P_0 >> 8) \quad (11)$$

20

【0103】

【数14】

$$P = (P_0 + P_1 + 1) >> 1 \quad (12)$$

【0104】

[0096] 短い遷移期間の場合、高速更新速度をもつ短距離予測（すなわち、より小さいウィンドウサイズ）のみが好ましい。しかし、最適値の近くでの安定化の後は、大部分のコンテキストについて 2 確率更新モデルがより正確である。JCTVC-F254 および JCTV-G764 は、最後の初期化からの更新のカウンタを導入することを提案する。あらゆる更新の後に、カウンタは 1 だけ増加する。カウンタが何らかのしきい値を超えるまで、式(10)によって定義される短い「ウィンドウサイズ」モデルのみが使用されることになる。カウンタがしきい値に達したとき、上記の式(12)によって定義されるより正確な 2 確率更新モデルに切り替えるべきである。JCTVC-F254 および JCTV-G764 によって提案された範囲計算プロセスは、 512×64 ルックアップテーブルを用いて実行される。 30

【0105】

[0097] JCTVC-F254 および JCTV-G764 によって提案された方法によれば、異なるコンテキスト初期化方法が適用される。詳細には、（それぞれ、`ascctxInit[0]` および `ascctxInit[1]` によって示される）2 つのパラメータが各コンテキストのためにあらかじめ定義され、初期確率状態 $iP0$ が、式(13)に示されているように導出される。 40

【0106】

【数15】

```
Int iQPreper = 1 slice ? 37 : 40;
Int c=ascCtxInit[0]+ascCtxInit[1]*(iQp - iQPreper);
iP0= min( max( 1, c ), 32767 );
```

【0107】

[0098] 1 確率更新モデルでは、コンテキストは、15 ビット精度で $iP0$ によって表される。2 確率更新モデルでは、別の変数 $iP1$ が最初に $iP0$ に等しく設定され、いくつのビンがコーディングされたかのカウンタがさらに必要とされる。JCTVC-F25 50

4 および J C T V - G 7 6 4 によって提案された方法では、 a s C t x I n i t [0] と a s C t x I n i t [1] の両方は 16 ビットに記憶される。

【 0 1 0 8 】

[0099] しかしながら、いくつかの例では、上記で説明された技法（すなわち、 H E V C の C A B A C 技法、および J C T V C - F 2 5 4 および J C T V - G 7 6 4 によって提案された修正）は、コーディング効率を低減し、および / またはコーダシステムリソースを準最適に利用し得る、1つまたは複数の問題を有し得る。

【 0 1 0 9 】

[0100] 一例として、（たとえば、 H E V C または H . 2 6 4 / A V C において使用される）上記で説明されたルックアップテーブルベースの算術コーダ技法では、確率更新は、固定ウィンドウサイズでの固定テーブル（すなわち、 T r a n s I d x L P S および T r a n s I d x M P S ）に基づく。固定ウィンドウサイズのこの使用により、更新速度が固定されることになる。しかしながら、シンタックス要素が発生し、コーディングされる必要がある頻度は、所与の C T U またはスライスについてまったく異なり得る。所与の C T U またはスライスについての異なる頻度で発生するシンタックス要素と組み合わせられた固定更新速度の制限により、より低い頻度で発生するシンタックス要素の推定された確率が準最適になり得る。たとえば、1つの C U について、 i n t e r _ p r e d _ i d c の最高 2 つの値がシグナリングされ得、1つの C U 内の変換係数が数回コーディングされ得る。この場合、これらのシンタックス要素について同じ更新速度を使用するとき、1に等しい i n t e r _ p r e d _ i d c の推定された確率は、変換係数の確率が比較的最適になっていることがあるにもかかわらず、1つのスライス全体をコーディングした後にまだ準最適であり得る。

【 0 1 1 0 】

[0101] 別の例として、（たとえば、 J C T V C - F 2 5 4 および J C T V - G 7 6 4 によって提案された）カウンタ技法に基づく上記で説明された算術コーダでは、確率更新速度（ probability update speed ）は固定であり、高い精度（たとえば、可能な確率インデックスが [1 , 2 ¹⁵ - 1] ）であり得るは、より低い頻度で選択されるシンタックス要素について低い効率を生じ、これは、望ましくないことがある。

【 0 1 1 1 】

[0102] 別の例として、カウンタ技法に基づく算術コーダの 2 確率更新モデル構成要素では、2つのステータスパラメータ（確率インデックス）が記憶され、更新されなければならず、これは、 C A B A C プロセスのスループットを望ましくなく制限し得る。

【 0 1 1 2 】

[0103] また別の例として、画像 / ビデオコーディングシステムでは、数百個のコンテキストが使用され得る。 J C T V C - F 2 5 4 および J C T V - G 7 6 4 によって提案された技法では、コンテキストごとに 32 ビットが必要とされるが、 H E V C における算術コーダには 8 ビットのみで十分である。したがって、 J C T V C - F 2 5 4 および J C T V - G 7 6 4 によって提案された技法におけるコンテキスト初期化のためのあらかじめ定義された値のストレージは 300 % だけ増加され、これは、ストレージに関するハードウェア実装形態について望ましくないことがある。

【 0 1 1 3 】

[0104] 本開示の 1 つまたは複数の技法によれば、ビデオコーダ（たとえば、ビデオエンコーダ 20 および / またはビデオデコーダ 30 ）は、第 1 のビット精度で記憶されたあらかじめ定義された初期化値から決定された初期確率状態から、第 2 のより大きいビット精度を有する初期確率状態に変換するために、ルックアップテーブルを使用し得る。たとえば、両方とも 16 ビット精度で記憶された 2 つのあらかじめ定義された初期化値（たとえば、 J C T V C - F 2 5 4 および J C T V - G 7 6 4 に記載されている a s C t x I n i t [0] および a s C t x I n i t [1] ）に基づいてコンテキストのための初期確率状態を決定することとは対照的に、ビデオコーダは、8 ビット精度で記憶された単一のあらかじめ定義された初期化値（たとえば、 H E V C において使用される i n i t V a l ）に

10

20

30

40

50

基づいてコンテキストのための中間初期確率状態を決定し、中間初期確率状態を、15ビット精度を有する初期確率状態（たとえば、JCTVC-F254およびJCTV-G764に記載されているIP0またはIP1）にマッピングするためにルックアップテーブルを使用し得る。いくつかの例では、ルックアップテーブルは、コンテキストのための可能な初期確率状態の数よりも少数のエントリを含んでいることがある。たとえば、32767個の可能な初期確率状態がある場合、ルックアップテーブルは128個のエントリを含み得る。このようにして、ビデオコーダは、（たとえば、HEVCと比較して）あらかじめ定義された初期化値のためのストレージ要件を増加させることなしに、（たとえば、HEVCと比較して）比較的多数の確率状態を有するコンテキストを初期化し得る。

【0114】

10

[0105] 本開示で説明される技法は、たとえば、ビデオエンコーダ、ビデオデコーダ、または組み合わせられたビデオエンコーダデコーダ（CODEC）内で実行され得る。特に、そのような技法は、ビデオエンコーダのエントロピー符号化ユニット（entropy encoding unit）および／またはビデオデコーダのエントロピー復号ユニット（entropy decoding unit）において実行され得る。本技法は、たとえば、HEVC規格の態様によるビデオコーディングなど、ビデオコーディングをサポートするように構成され得るCABACプロセス内で実行され得るエントロピー符号化および復号ユニットは、たとえば、残差ビデオデータに関連する量子化された変換係数、動きベクトル情報、シンタックス要素、ならびにビデオ符号化および／またはビデオ復号プロセスにおいて有用であり得る他のタイプの情報など、様々なビデオデータのうちのいずれかを符号化または復号するために、相反するまたは逆の様式でコーディングプロセスを適用するであり得る。

【0115】

20

[0106] 図4は、本開示で説明される、BACコーディングのための技法を利用するように構成され得るビデオエンコーダ20の一例を示すブロック図である。ビデオエンコーダ20は、例示のためにHEVCコーディングのコンテキストにおいて説明されるが、他のコーディング規格または方法に関して本開示を限定するものではない。その上、ビデオエンコーダ20は、HEVCの範囲拡張（range extension）に従って技法を実装するように構成され得る。

【0116】

30

[0107] ビデオエンコーダ20は、ビデオスライス内のビデオブロックのイントラコーディングおよびインターコーディングを実行し得る。イントラコーディングは、所与のビデオピクチャ内のビデオの空間冗長性を低減または除去するために空間予測に依拠する。インターコーディングは、ビデオシーケンスの隣接するピクチャ内のビデオの時間冗長性を低減または除去するか、あるいは他のビュー中のビデオに関する冗長性を低減または除去するために、時間予測またはビュー間予測に依拠する。

【0117】

40

[0108] 図4の例では、ビデオエンコーダ20は、ビデオデータメモリ40と、予測処理ユニット42と、参照ピクチャメモリ64と、加算器50と、変換処理ユニット52と、量子化処理ユニット54と、エントロピー符号化ユニット（entropy encoding unit）56とを含む。予測処理ユニット42は、動き推定ユニット44と、動き補償ユニット46と、イントラ予測ユニット48とを含む。ビデオブロック再構成のために、ビデオエンコーダ20はまた、逆量子化処理ユニット58と、逆変換処理ユニット60と、加算器62とを含む。再構成されたビデオからブロック性アーティファクト（blockiness artifact）を除去するためにブロック境界をフィルタ処理するための（図4に示されていない）デブロッキングフィルタも含まれ得る。所望される場合、デブロッキングフィルタは、一般に、加算器62の出力をフィルタ処理することになる。（ループ中またはループ後の）追加のループフィルタもデブロッキングフィルタに加えて使用され得る。

【0118】

[0109] ビデオデータメモリ40は、ビデオエンコーダ20の構成要素によって符号化されるべきビデオデータを記憶し得る。ビデオデータメモリ40に記憶されるビデオデータ

50

タは、たとえば、ビデオソース 18 から取得され得る。参照ピクチャメモリ 64 は、(たとえば、イントラ予測コーディングモードまたはインター予測コーディングモードとも呼ばれる、イントラコーディングモードまたはインターコーディングモードで)ビデオエンコーダ 20 によってビデオデータを符号化する際に使用するための参照ビデオデータを記憶する復号ピクチャバッファ (D P B : decoded picture buffer) の一例である。ビデオデータメモリ 40 および参照ピクチャメモリ 64 は、同期 D R A M (S D R A M) を含むダイナミックランダムアクセスメモリ (D R A M) 、磁気抵抗 R A M (M R A M) 、抵抗性 R A M (R R A M (登録商標)) 、または他のタイプのメモリデバイスなど、様々なメモリデバイスのうちのいずれかによって形成され得る。ビデオデータメモリ 40 および参照ピクチャメモリ 64 は、同じメモリデバイスまたは別個のメモリデバイスによって与えられ得る。様々な例では、ビデオデータメモリ 40 は、ビデオエンコーダ 20 の他の構成要素とともにオンチップであるか、またはそれらの構成要素に対してオフチップであり得る。

【 0 1 1 9 】

[0110] 符号化プロセス中に、ビデオエンコーダ 20 は、コーディングされるべきビデオピクチャまたはスライスを受信する。ピクチャまたはスライスは複数のビデオブロックに分割され得る。動き推定ユニット 44 および動き補償ユニット 46 は、時間圧縮を行うかまたはビュー間圧縮を行うために、1つまたは複数の参照ピクチャ中の1つまたは複数のブロックに対する受信されたビデオブロックのインター予測コーディングを実行する。イントラ予測ユニット 48 は、代替的に、空間圧縮を行うために、コーディングされるべきブロックと同じピクチャまたはスライス中の1つまたは複数の隣接ブロックに対する受信されたビデオブロックのイントラ予測コーディングを実行し得る。ビデオエンコーダ 20 は、(たとえば、ビデオデータのブロックごとに適切なコーディングモードを選択するために)複数のコーディングパスを実行し得る。

【 0 1 2 0 】

[0111] その上、パーティションユニット (図示せず) が、前のコーディングパスにおける前の区分方式の評価に基づいて、ビデオデータのブロックをサブブロックに区分し得る。たとえば、パーティションユニットは、初めにピクチャまたはスライスを L C U に区分し、レートひずみ分析 (たとえば、レートひずみ最適化) に基づいて L C U の各々をサブ C U に区分し得る。予測処理ユニット 42 は、さらに、サブ C U への L C U の区分を示す4分木データ構造を生成し得る。4分木のリーフノード C U は、1つまたは複数の P U と1つまたは複数の T U とを含み得る。

【 0 1 2 1 】

[0112] 予測処理ユニット 42 は、たとえば、誤差結果に基づいてコーディングモード、すなわち、イントラまたはインターのうちの1つを選択し得、残差ブロックデータを生成するために、得られたイントラコード化ブロックまたはインターコード化ブロックを加算器 50 に与え、参照ピクチャとして使用するための符号化ブロックを再構成するために、得られたイントラコード化ブロックまたはインターコード化ブロックを加算器 62 に与える。予測処理ユニット 42 はまた、動きベクトル、イントラモードインジケータ、パーティション情報、および他のそのようなシンタックス情報など、シンタックス要素をエンタロピー符号化ユニット 56 に与える。

【 0 1 2 2 】

[0113] 動き推定ユニット 44 と動き補償ユニット 46 とは、高度に統合され得るが、概念的な目的のために別々に示されている。動き推定ユニット 44 によって実行される動き推定は、ビデオブロックの動きを推定する動きベクトルを生成するプロセスである。動きベクトルは、たとえば、現在ピクチャ (または他のコード化ユニット) 内でコーディングされている現在ブロックに対する参照ピクチャ (または他のコード化ユニット) 内の予測ブロックに対する現在ビデオピクチャ内のビデオブロックの P U の変位を示し得る。予測ブロックは、絶対差分和 (S A D : sum of absolute difference) 、2乗差分和 (S S D : sum of square difference) 、または他の差分メトリックによって決定され得るピク

10

20

30

40

50

セル差分に関して、コーディングされるべきブロックにぴったり一致することがわかるブロックである。いくつかの例では、ビデオエンコーダ20は、参照ピクチャメモリ64に記憶された参照ピクチャのサブ整数ピクセル位置の値を計算し得る。たとえば、ビデオエンコーダ20は、参照ピクチャの1/4ピクセル位置、1/8ピクセル位置、または他の分数ピクセル位置の値を補間し得る。したがって、動き推定ユニット44は、フルピクセル位置と分数ピクセル位置とに対して動き探索を実行し、分数ピクセル精度で動きベクトルを出力し得る。

【0123】

[0114] 動き推定ユニット44は、PUの位置を参照ピクチャの予測ブロックの位置と比較することによって、インターフォード化スライス中のビデオブロックのPUのための動きベクトルを計算する。参照ピクチャは、参照ピクチャメモリ64に記憶された1つまたは複数の参照ピクチャを識別する1つまたは複数の参照ピクチャリスト(RPL)から選択され得る。動き推定ユニット44は、計算された動きベクトルをエントロピー符号化ユニット56と動き補償ユニット46とに送る。いくつかの例では、動き推定ユニット44は、選択された参照ピクチャの指示をエントロピー符号化ユニット56に送り得る。

【0124】

[0115] 動き補償ユニット46によって実行される動き補償は、動き推定ユニット44によって決定された動きベクトルに基づいて予測ブロックをフェッチまたは生成することを伴い得る。同じく、動き推定ユニット44および動き補償ユニット46は、いくつかの例では、機能的に統合され得る。現在ビデオブロックのPUのための動きベクトルを受信すると、動き補償ユニット46は、動きベクトルが参照ピクチャリスト(RPL: reference picture list)のうちの1つにおいて指す予測ブロックの位置を特定し得る。加算器50は、以下で説明されるように、コーディングされている現在ブロックのピクセル値から予測ブロックのピクセル値を減算し、ピクセル差分値を形成することによって、残差ビデオブロックを形成する。概して、動き推定ユニット44はルーマ成分に対して動き推定を実行し、動き補償ユニット46は、クロマ成分とルーマ成分の両方のためにルーマ成分に基づいて計算された動きベクトルを使用する。予測処理ユニット42はまた、ビデオスライスのビデオブロックを復号する際にビデオデコーダ30が使用するためのビデオブロックとビデオスライスとに関連するシンタックス要素を生成し得る。

【0125】

[0116] イントラ予測ユニット48は、上記で説明されたように、動き推定ユニット44と動き補償ユニット46とによって実行されるインター予測の代替として、現在ブロックをイントラ予測し得る。特に、イントラ予測ユニット48は、現在ブロックを符号化するために使用すべきイントラ予測モードを決定し得る。いくつかの例では、イントラ予測ユニット48は、たとえば、別個の符号化バス中に、様々なイントラ予測モードを使用してブロックを符号化し得、イントラ予測ユニット48は、複数のイントラ予測モードから使用するのに適切なイントラ予測モードを選択し得る。

【0126】

[0117] たとえば、イントラ予測ユニット48は、様々なテストされたイントラ予測モードのためのレートひずみ分析(rate-distortion analysis)を使用してレートひずみ値を計算し、テストされたモードの中で最良のレートひずみ特性を有するイントラ予測モードを選択し得る。レートひずみ分析は、概して、符号化ブロックと、符号化ブロックを生成するために符号化された元の符号化されていないブロックとの間のひずみ(または誤差)の量、ならびに符号化ブロックを生成するために使用されるビットレート(すなわち、ビット数)を決定する。イントラ予測ユニット48は、どのイントラ予測モードがブロックについて最良のレートひずみ値を呈するかを決定するために、様々な符号化ブロックのためのひずみおよびレートから比を計算し得る。いくつかの例では、複数のイントラ予測モードの各々は、イントラ予測ユニット48によって(すなわち、ビデオデコーダに)シグナリングされ得る、対応するモードインデックスを有し得る。

【0127】

10

20

30

40

50

[0118] ビデオエンコーダ 20 は、コーディングされている元のビデオブロックから、予測処理ユニット 42 からの予測データを減算することによって残差ビデオブロックを形成する。加算器 50 は、この減算演算を実行する 1 つまたは複数の構成要素を表す。

【 0 1 2 8 】

[0119] 変換処理ユニット 52 は、離散コサイン変換 (DCT) または概念的に同様の変換などの変換を残差ブロックに適用し、残差変換係数値を備えるビデオブロックを生成する。変換処理ユニット 52 は、DCT と概念的に同様である他の変換を実行し得る。ウェーブレット変換、整数変換、サブバンド変換または他のタイプの変換も使用され得る。いずれの場合も、変換処理ユニット 52 は、変換を残差ブロックに適用し、残差変換係数のブロックを生成する。変換は、残差情報をピクセル値領域から周波数領域などの変換領域に変換し得る。

【 0 1 2 9 】

[0120] 変換処理ユニット 52 は、得られた変換係数を量子化処理ユニット 54 に送り得る。量子化処理ユニット 54 は、ビットレートをさらに低減するために変換係数を量子化する。量子化プロセスは、係数の一部または全部に関連するビット深度を低減し得る。量子化の程度は、量子化パラメータ (quantization parameter) を調整することによって修正され得る。いくつかの例では、量子化処理ユニット 54 は、次いで、量子化された変換係数を含む行列の走査を実行し得る。代替的に、エントロピー符号化ユニット 56 が走査を実行し得る。

【 0 1 3 0 】

[0121] 変換係数が 1 次元アレイに走査されると、エントロピー符号化ユニット 56 は、コンテキスト適応型可変長コーディング (CAVLC)、コンテキスト適応型バイナリ算術コーディング (CABAC)、確率間隔区分エントロピーコーディング (PIPE)、ゴロムコーディング、ゴロム - ライスコーディング、指数ゴロムコーディング、シンタクスベースコンテキスト適応型バイナリ算術コーディング (SBA : syntax-based context-adaptive binary arithmetic coding)、または別のエントロピーコーディング方法などのエントロピーコーディングを係数に適用し得る。本開示の例による、様々な異なるエントロピーコーディングプロセスへの参照が行われるが、エントロピー符号化ユニット 56 は、上記で説明されたように BAC コーディングを実行するように構成され得る。

【 0 1 3 1 】

[0122] CAVLC を実行するために、エントロピー符号化ユニット 56 は、送信されるべきシンボルのための可変長コードを選択し得る。VLC 中のコードワードは、比較的より短いコードが、可能性がより高いシンボルに対応し、より長いコードが、可能性がより低いシンボルに対応するように構成され得る。このようにして、VLC の使用は、たとえば、送信されるべき各シンボルのための等長コードワードを使用することに勝るビット節約を達成し得る。

【 0 1 3 2 】

[0123] CABAC を実行するために、エントロピー符号化ユニット 56 は、送信されるべきシンボルを符号化するために、あるコンテキストに適用すべきコンテキストを選択し得る。コンテキストは、たとえば、隣接値が非 0 であるか否かに関係し得る。エントロピー符号化ユニット 56 はまた、選択された変換を表す信号など、シンタクス要素をエントロピー符号化し得る。エントロピー符号化ユニット 56 によるエントロピーコーディングの後に、得られた符号化ビデオは、ビデオデコーダ 30 などの別のデバイスに送信されるか、あるいは後で送信するかまたは取り出すためにアーカイブされ得る。

【 0 1 3 3 】

[0124] 本開示の 1 つまたは複数の技法によれば、エントロピー符号化ユニット 56 は、ビデオデータを復号する際に、ビデオデコーダ 30 など、ビデオデコーダが使用するためのデータ (たとえば、1 次元バイナリベクトルとして表されるシンタクス要素値) をエントロピー符号化 (entropy encode) するとき、異なるウィンドウサイズを選択し得る。エントロピー符号化ユニット 56 の一例のさらなる詳細は、図 5 を参照しながら以下で

10

20

30

40

50

説明される。

【0134】

[0125] 逆量子化処理ユニット58および逆変換処理ユニット60は、たとえば、参照ブロックとして後で使用するために、ピクセル領域において残差ブロックを再構成するために、それぞれ逆量子化および逆変換を適用する。

【0135】

[0126] 動き補償ユニット46はまた、動き推定において使用するためのサブ整数ピクセル値を計算するために、参照ブロックに1つまたは複数の補間フィルタを適用し得る。加算器62は、参照ピクチャメモリ64に記憶するための再構成されたビデオブロックを生成するために、動き補償ユニット46によって生成された動き補償予測ブロックに再構成された残差ブロックを加算する。再構成されたビデオブロックは、後続のビデオピクチャ中のブロックをインターフォーディングするために動き推定ユニット44および動き補償ユニット46によって参照ブロックとして使用され得る。現在ピクチャが現在ピクチャを予測するための参照ピクチャとして使用される場合など、いくつかの例では、動き補償ユニット46および/または加算器62は、現在ピクチャをコーディングしながら、一定の間隔で、参照ピクチャメモリ64によって記憶された現在ピクチャのバージョンを更新し得る。一例として、動き補償ユニット46および/または加算器62は、現在ピクチャの各ブロックをコーディングした後に、参照ピクチャメモリ64によって記憶された現在ピクチャのバージョンを更新し得る。たとえば、現在ブロックのサンプルが、初期化された値として参照ピクチャメモリ64に記憶される場合、動き補償ユニット46および/または加算器62は、現在ブロックのための再構成されたサンプルで、参照ピクチャメモリ64によって記憶された現在ピクチャの現在のサンプルを更新し得る。

10

【0136】

[0127] フィルタ処理ユニット(図示せず)は、様々なフィルタ処理プロセスを実行し得る。たとえば、フィルタ処理ユニットはデブロッキングを実行し得る。すなわち、フィルタ処理ユニットは、再構成されたビデオのスライスまたはフレームを形成する複数の再構成されたビデオブロックを受信し、スライスまたはフレームからプロッキネスアーティファクトを除去するために、ブロック境界をフィルタ処理し得る。一例では、フィルタ処理ユニットは、ビデオブロックのいわゆる「境界強度(boundary strength)」を評価する。ビデオブロックの境界強度に基づいて、ビデオブロックのエッジピクセルが、閲覧者にとって1つのビデオブロックからの遷移を知覚することがより困難になるように、隣接するビデオブロックのエッジピクセルに対してフィルタ処理され得る。

20

【0137】

[0128] いくつかの例では、動き補償ユニット46および/または加算器62は、フィルタ処理がサンプルに対するフィルタ処理(たとえば、デブロッキングおよび/またはSAD)を実行する前に、参照ピクチャメモリ64によって記憶された現在ピクチャのバージョンを更新し得る。たとえば、フィルタ処理ユニットは、フィルタ処理を適用する前に、ピクチャ全体がコーディングされるまで待ち得る。このようにして、動き推定ユニット44は、フィルタ処理を適用する前に、参照として現在ピクチャを使用し得る。いくつかの例では、フィルタ処理ユニットは、参照ピクチャメモリ64によって記憶された現在ピクチャのバージョンが更新されたとき、フィルタ処理を実行し得る。たとえば、フィルタ処理ユニットは、各ブロックが更新されたとき、フィルタ処理を適用し得る。このようにして、動き推定ユニット44は、フィルタ処理を適用した後、参照として現在ピクチャを使用し得る。

30

【0138】

[0129] 本技法のいくつかの異なる態様および例が本開示で説明されるが、本技法の様々な態様および例は、一緒にまたは互いに別々に実行され得る。言い換えれば、本技法は、上記で説明された様々な態様および例に厳密に限定されるべきではなく、組み合わせて使用されるか、あるいは一緒におよび/または別々に実行され得る。さらに、いくつかの技法は、(イントラ予測ユニット48、動き補償ユニット46、またはエントロピー符号

40

50

化ユニット 5 6 などの) ビデオエンコーダ 2 0 のいくつかのユニットに起因され得るが、ビデオエンコーダ 2 0 の 1 つまたは複数の他のユニットも、そのような技法を行うことを担当し得ることを理解されたい。

【 0 1 3 9 】

[0130] 図 5 は、本開示の技法による、 C A B A C を実行するように構成され得る例示的なエントロピー符号化ユニット 5 6 のブロック図である。シンタックス要素 1 1 8 がエントロピー符号化ユニット (entropy encoding unit) 5 6 に入力される。シンタックス要素がすでにバイナリ値シンタックス要素 (たとえば、フラグ、または 0 および 1 の値のみを有する他のシンタックス要素) である場合、 2 値化のステップはスキップされ得る。シンタックス要素が非バイナリ値シンタックス要素 (non-binary valued syntax element) (たとえば、 1 または 0 以外の値を有し得るシンタックス要素) である場合、 非バイナリ値シンタックス要素はバイナライザ 1 2 0 によって 2 値化される。バイナライザ 1 2 0 は、バイナリ決定のシーケンスへの非バイナリ値シンタックス要素のマッピングを実行する。これらのバイナリ決定は、しばしば「ビン」と呼ばれる。たとえば、変換係数レベルでは、レベルの値は連続するビンに分けられ得、各ビンは、係数レベルの絶対値がある値よりも大きいか否かを示す。たとえば、(有意性フラグと呼ばれることがある) ビン 0 は、変換係数レベルの絶対値が 0 よりも大きいか否かを示す。ビン 1 は、変換係数レベルの絶対値が 1 よりも大きいか否かを示す、などである。各非バイナリ値シンタックス要素について、一意のマッピングが作成され得る。

【 0 1 4 0 】

[0131] バイナライザ 1 2 0 によって生成された各ビンは、エントロピー符号化ユニット 5 6 のバイナリ算術コーディング側に供給される。すなわち、非バイナリ値シンタックス要素の所定のセットについて、各ビンタイプ (たとえば、ビン 0) が次のビンタイプ (たとえば、ビン 1) の前にコーディングされる。コーディングは、通常モードまたはバイパスモードのいずれかで実行され得る。バイパスモードでは、バイパスコーディングエンジン 1 2 6 が、固定確率モデルを使用して、たとえば、ゴロム - ライスまたは指数ゴロムコーディングを使用して、算術コーディングを実行する。バイパスモードは、概して、より予測可能なシンタックス要素のために使用される。

【 0 1 4 1 】

[0132] 通常モードでのコーディングは、 C A B A C を実行することを伴う。通常モード C A B A C は、ビンの値の確率が、前にコーディングされたビンのそのときの値を与られれば予測可能である場合に、ビン値をコーディングするためのものである。ビンが L P S である確率がコンテキストモデル (context modeler) 1 2 2 によって決定される。コンテキストモデル 1 2 2 は、ビン値とコンテキストのための確率状態 (たとえば、 L P S の値と、 L P S が発生する確率とを含む確率状態) とを出力する。コンテキストは、一連のビンのための初期コンテキストであり得るか、または前にコーディングされたビンのコード化値に基づいて決定され得る。上記で説明されたように、コンテキストモデル 1 2 2 は、受信されたビンが M P S であったのか L P S であったのか否かに基づいて状態を更新し得る。コンテキストおよび確率状態 がコンテキストモデル 1 2 2 によって決定された後、通常コーディングエンジン 1 2 4 がビン値に対して B A C を実行する。

【 0 1 4 2 】

[0133] 本開示の 1 つまたは複数の技法によれば、バイナリ算術コーディングプロセスにおいて確率状態を更新するために使用される変数の同じ値 (たとえば、 ウィンドウサイズ、スケーリングファクタ () 、および確率更新速度のうちの 1 つまたは複数) を使用することとは対照的に、エントロピー符号化ユニット 5 6 は、異なるコンテキストおよび / または異なるシンタックス要素について変数の異なる値を使用し得る。たとえば、コンテキストモデル 1 2 2 は、複数のコンテキストのうちのコンテキストについて、バイナリ算術コーディングプロセスにおいて確率状態を更新するために使用される変数の値を決定し、決定された値に基づいて確率状態を更新し得る。 ウィンドウサイズは、更新された確率状態に影響を及ぼす、前にコーディングされたビンの数を表し得る。

10

20

30

40

50

【0143】

[0134] いくつかの例では、次の確率状態を決定するためにコンテキストモデル122によって使用されるウィンドウサイズはコンテキスト依存にされ得る。たとえば、コンテキストモデル122は、異なるコンテキストについて異なるウィンドウサイズを使用し得る。一例として、コンテキストモデル122は、複数のコンテキストのうちの第1のコンテキストのための第1のウィンドウサイズを決定し、第1のウィンドウサイズとは異なる、複数のコンテキストのうちの第2のコンテキストのための第2のウィンドウサイズを決定し得る。

【0144】

[0135] いくつかの例では、上記のコンテキストモデル依存更新方法を、JCTVC-F254およびJCTVC-G764におけるものなど、カウンタベース算術コーダに組み込むとき、ウィンドウサイズの値はコンテキストに依存し得る。さらに、各コンテキストは、式(4)からの確率 P_i に加えて、ウィンドウサイズにさらに関連し得る。

10

【0145】

[0136] いくつかの例では、コンテキストモデル122は、 2^M に等しいことがあるウィンドウサイズ W を使用し得、ここで、 M は正の整数であり得る。したがって、いくつかのコンテキストは同じ M 値を有し得るが、各コンテキストは、他のコンテキストとは異なり得る、それ自体の M 値を有し得る。

【0146】

[0137] いくつかの例では、コンテキストモデル122は、ウィンドウサイズのあらかじめ定義されたセット(pre-defined set)からウィンドウサイズを決定し得る。いくつかの例示的なあらかじめ定義されたウィンドウサイズは、16、32、64、および128であるが、他のウィンドウサイズが企図される。たとえば、可能な M 値のセットがあらかじめ定義され得、たとえば、 M は、両端値を含む、4から7までにわたることがある。いくつかの例では、コンテキストモデル122は、可能なウィンドウサイズのセットの指示(たとえば、可能な M 値のセットの指示)が、スライスヘッダ、あるいはピクチャパラメータセット、アクティブパラメータセット、シーケンスパラメータセット、またはビデオパラメータセットを含むパラメータセット中でシグナリングされることを引き起こし得る。

20

【0147】

[0138] いくつかの例では、各コンテキストに関連するウィンドウサイズ(たとえば、 M の値)はあらかじめ定義され得る。いくつかの例では、ウィンドウサイズは、さらに、スライスタイプおよび/または(たとえば、HEVCではtemporal IDと呼ばれる)時間識別子に依存し得る。いくつかの例では、ウィンドウサイズは、さらに、ピクチャタイプ(またはNALユニットタイプ)、たとえば、ピクチャがランダムアクセスピクチャであるか否かに依存し得る。

30

【0148】

[0139] いくつかの例では、コンテキストモデル122は、各コンテキストに関連するウィンドウサイズ(たとえば、 M の値)が、スライスヘッダ/ピクチャパラメータセット/アクティブパラメータセット/シーケンスパラメータセット中でなど、ビットストリーム中でシグナリングされることを引き起こし得る。たとえば、各コンテキストのためのデフォルトウィンドウサイズ(default window size)が最初にあらかじめ定義され得る。各それぞれのコンテキストについて、コンテキストモデル122は、デフォルトウィンドウサイズがそれぞれのコンテキストのために使用されるかどうかを示す、それぞれのシンタックス要素(たとえば、フラグ)を符号化し得る。デフォルトウィンドウサイズがそれぞれのコンテキストのために使用されない場合、コンテキストモデル122は、デフォルトウィンドウサイズに基づいて、実際の使用されるウィンドウサイズを差分符号化し得る。いくつかの例では、コンテキストモデル122は、すべてのコンテキストの(すなわち、デフォルトウィンドウサイズが使用されるかどうかを示す)シンタックス要素と一緒に編成し、これらのシンタックス要素をコーディングするためにランレンジングコーディング

40

50

を利用し得る。いくつかの例では、コンテキストモデル 122 は、実際に使用されるウィンドウサイズとデフォルトウィンドウサイズとの間の差分をコーディングするとき、マッピングテーブルを利用し得る。たとえば、デフォルトの M 値が 6 に等しい場合、可能な M 値は、4、5、6、および 7 である。マッピングテーブル (mapping table) は次のように定義され得る。

【0149】

【表 6】

実際の M 値	4	5	6	7
コーディングされるべき値	0	1	-	2

10

【0150】

[0140] いくつかの例では、コンテキストモデル 122 は、各コンテキストについて実際のウィンドウサイズとデフォルトウィンドウサイズとの間の差分を直接コーディングし得る。たとえば、デフォルトの M 値が 4 である場合、コンテキストモデル 122 は、各コンテキストについて M - 4 をコーディングし得る。

【0151】

[0141] いくつかの例では、コンテキストモデル 122 は、現在スライス (current slice) 中のコンテキストのためのすべてのウィンドウサイズが、前にコーディングされたスライス中の対応するコンテキストのための継承 (inherit) されたウィンドウサイズである（すなわち、前にコーディングされたスライス中の対応するコンテキストのためのウィンドウサイズに等しく設定される）かどうかを示す、第 1 のシンタックス要素をコーディングし得る。一例では、「前に復号されたスライス (previously decoded slice)」は、現在スライスと同じスライスタイプ、または同じスライスタイプと量子化パラメータの両方、または同じスライスタイプと時間レイヤの両方、および / あるいは同じ初期化された量子化パラメータを有する、前にコーディングされたスライスとして定義され得る。いくつかの例では、前のスライスは、D P B 中に存在するピクチャに属することが必要であり得、参照ピクチャとして現在ピクチャのために使用され得、特に、HEVC ベースプラットフォームの場合のように、前のスライスは、参照ピクチャセット (RPS) 中のピクチャ、さらには、RPS の以下のサブセット、すなわち、RefPicSetStCurrBefore、RefPicSetStCurrAfter、および RefPicSetLtCurr のうちの 1 つ中のピクチャに属することが必要とされ得る。

20

【0152】

[0142] いくつかの例では、コンテキストモデル 122 は、デフォルトウィンドウサイズが（たとえば、現在スライス中の）複数のコンテキストのために使用されるかどうかを示す第 1 のシンタックス要素をコーディングし得る。デフォルトウィンドウサイズが複数のコンテキストのために使用されない場合、コンテキストモデル 122 は、コンテキストのためにウィンドウサイズを示す第 2 のシンタックス要素をコーディングし得る。たとえば、コンテキストモデル 122 は、コンテキストのためにウィンドウサイズとデフォルトウィンドウサイズとの間の差分を示す第 2 のシンタックス要素をコーディングし得る。

30

【0153】

[0143] 別の例では、コンテキストモデル 122 は、たとえば、前のスライスまたはピクチャからのコード化情報に基づいて、ウィンドウサイズを導出し得る。

40

【0154】

[0144] いくつかの例では、算術コーダにおいて次の確率状態または確率更新速度 (probability update speed) を決定するために使用される「ウィンドウサイズ (window size)」は、たとえば、コンテキストが、異なるシンタックス要素の間で共有される場合、シンタックス要素固有であり得る。たとえば、シンタックス要素のビンを符号化するためにコンテキストを使用するとき、コンテキストモデル 122 は、シンタックス要素に基づいてコンテキストのためにウィンドウサイズを決定し得る。一例として、inter_pred_idc シンタックス要素のビンをコーディングするときにコンテキストの状態を

50

更新するために使用されるウィンドウサイズは、16（たとえば、 $M = 4$ ）であり得、`c o e f f_a b s_l e v e l_g r e a t e r 1_f l a g s`シンタックス要素のビンをコーディングするときにコンテキストの状態を更新するために使用されるウィンドウサイズは、128（たとえば、 $M = 7$ ）であり得る。

【0155】

[0145] 本開示の1つまたは複数の技法によれば、コンテキストモデル122は、ビデオデータを復号する際にビデオデコーダ30が使用するためのデータ（たとえば、1次元ベクトルを表すシンタックス要素および／または他のシンタックス要素）をエントロピー符号化するとき、異なるウィンドウサイズを適応的に決定し得る。たとえば、各コンテキストについて、コンテキストモデル122は、異なるウィンドウサイズで、記録されたビンストリングをコーディングするビットを計算し、最小ビットをもつ1つのウィンドウサイズを選択し得る。ウィンドウサイズがウィンドウサイズのあらかじめ定義されたセットから選択される場合、コンテキストモデル122は、ウィンドウサイズのあらかじめ定義されたセットのうちのそれぞれのウィンドウサイズについて、コンテキストを用いてビンストリングを符号化するために使用されるビットのそれぞれの量を決定し、ビットの最も小さい量に対応する、ウィンドウサイズのあらかじめ定義されたセットのうちのウィンドウサイズを、そのコンテキストのためのウィンドウサイズとして選択し得る。10

【0156】

[0146] いくつかの例では、上記の（1つまたは複数の）技法は特定のコンテキストに適用可能であり得る。すなわち、コンテキストのサブセットは、デフォルトウィンドウサイズではなく更新された「ウィンドウサイズ」を使用し得る。いくつかの例では、上記の（1つまたは複数の）技法は特定のスライスタイプに適用可能であり得る。20

【0157】

[0147] 上記で説明されたように、コンテキストモデル122は、コンテキストの確率状態を周期的に初期化し得る。たとえば、コンテキストモデル122は、ビデオデータの各スライスの開始時にコンテキストの確率状態を初期化し得る。コンテキストモデル122は、後続の確率状態がそこから導出され得る初期確率状態を決定することによって、コンテキストの確率状態を初期化し得る。（すなわち、HEVCの9.3.2.2に記載されている）HEVC初期化プロセスでは、コンテキストモデル122は、（たとえば、HEVCの表9-5～表9-31を使用して）どのシンタックス要素のどのビンが符号化をもたらすであるかに基づいて、あらかじめ定義された初期化値（たとえば、`i n i t V a l u e`）を取得し得る。あらかじめ定義された初期化値はビデオエンコーダ20によって記憶され得る。HEVCでは、あらかじめ定義された初期化値は8ビット精度で各々記憶される。30

【0158】

[0148] コンテキストモデル122は、取得されたあらかじめ定義された初期化値に基づいて初期確率状態を決定し得る。たとえば、コンテキストモデル122は、次いで、取得されたあらかじめ定義された初期化値に基づいて傾斜値（slope value）とオフセット値（offset value）とを決定し得る。いくつかの例では、コンテキストモデル122は、上記の式（3）またはHEVCの式（9-4）に従って傾斜値とオフセット値とを決定し得る。40

【0159】

[0149] コンテキストモデル122は、中間値（intermediate value）を決定するために、決定された傾斜値とオフセット値とを使用し得る。いくつかの例では、コンテキストモデル122は、`i n i t S t a t e`が中間値である、上記の式（3）、または`p r e C t x S t a t e`が中間値である、HEVCの式（9-6）に従って中間値を決定し得る。

【0160】

[0150] HEVCでは、コンテキストモデル122は、中間値に基づいて初期確率状態を決定し得る。たとえば、コンテキストモデル122は、状態インデックスが初期確率状態である、上記の式（3）、または`p S t a t e I d x`が初期確率状態である、HEVC

の式(9-6)に従って、中間値に基づいて初期確率状態を決定し得る。

【0161】

[0151] HEVCは128個のコンテキスト状態をサポートし、その各々は、MPSの値と確率状態の値とによって完全に定義され得る。MPSの値は1ビット値として記憶され得、確率状態は7ビット値として記憶され得、合計8ビットがコンテキストの現在状態を定義する。したがって、8ビットのあらかじめ定義された初期化値を使用することは、コンテキストモデル122が128個の可能な状態のうちのいずれかに初期化することを可能にするのに十分な情報を与える。

【0162】

[0152] 上記で説明されたように、JCTVC-F254およびJCTV-G764によって提案された方法は、異なる初期化技法を使用する。詳細には、JCTVC-F254およびJCTV-G764によって提案された方法では、(それぞれ、`ascctxInit[0]`および`ascctxInit[1]`によって示される)2つのパラメータが各コンテキストのためにあらかじめ定義され、初期確率状態`IP0`が、式(13)において上記で示されているように導出される。JCTVC-F254およびJCTV-G764によって提案された方法では、コンテキストは32767個の可能な状態を有し得る。したがって、JCTVC-F254およびJCTV-G764は、32767個の可能な状態のうちのいずれかへの初期化を可能にするために、あらかじめ定義された初期化値(すなわち、`ascctxInit[0]`および`ascctxInit[1]`)の両方が16ビット精度で記憶されることを提案した。したがって、JCTVC-F254およびJCTV-G764によって提案された方法は、あらかじめ定義された初期化値を記憶するために必要とされるメモリの量の300%の増加を生じ、これは、ハードウェア実装形態について望ましくないことがある。

【0163】

[0153] 本開示の1つまたは複数の技法によれば、コンテキストモデル122は、2のN乗よりも大きい数の可能な確率状態を有するコンテキストを初期化するために、Nビット精度で記憶されたあらかじめ定義された初期化値を利用し得る。たとえば、コンテキストモデル122は、あらかじめ定義された初期化値に基づいて中間値を決定し、中間値をコンテキストのための初期確率状態に変換するためにルックアップテーブルを使用し得る。いくつかの例では、ルックアップテーブル中に含まれるエントリの数は2のN乗に等しいことがある。そのような例では、コンテキストモデル122は、コンテキストの可能な確率状態のサブセットに初期化することのみが可能であり得る。しかしながら、あらかじめ定義された初期化値のストレージ要件を低減することによって得られる利益は、可能な確率状態の完全な範囲に初期化することができないことから生じる精度低減よりも大きいことがある。

【0164】

[0154] 1つの例示的な例として、`qp`によって示される入力量子化パラメータと、`initVal`によって示されるあらかじめ定義された値とが与えられれば、コンテキストモデル122は、(状態インデックスによって示される)確率モデルの15ビットエントリを以下のように導出し得る。

【0165】

10

20

30

40

【数16】

```

qp      = Clip3(0, 51, qp);
傾斜    = (initVal >> 4) * 5 - 45;
オフセット = ((initVal & 15) << 3) - 16;
initState = min( max(1, ((傾斜 * qp) >> 4) + オフセット), 126 );
状態インデックス = m_MappedProb[initState];

```

上式で m_MappedProb[128] =

```

{
  614, 647, 681, 718, 756, 797, 839, 884, 932, 982, 1034, 1089, 1148, 1209, 1274,
  1342, 1414, 1490, 1569, 1653, 1742, 1835, 1933, 2037, 2146, 2261, 2382, 2509, 2643,
  2785, 2934, 3091, 3256, 3430, 3614, 3807, 4011, 4225, 4452, 4690, 4941, 5205, 5483,
  5777, 6086, 6412, 6755, 7116, 7497, 7898, 8320, 8766, 9235, 9729, 10249, 10798,
  11375, 11984, 12625, 13300, 14012, 14762, 15551, 16384, 16384, 17216, 18005,
  18755, 19467, 20142, 20783, 21392, 21969, 22518, 23038, 23532, 24001, 24447,
  24869, 25270, 25651, 26012, 26355, 26681, 26990, 27284, 27562, 27826, 28077,
  28315, 28542, 28756, 28960, 29153, 29337, 29511, 29676, 29833, 29982, 30124,
  30258, 30385, 30506, 30621, 30730, 30834, 30932, 31025, 31114, 31198, 31277,
  31353, 31425, 31493, 31558, 31619, 31678, 31733, 31785, 31835, 31883, 31928,
  31970, 32011, 32049, 32086, 32120, 32153,
};

```

【0166】

[0155] 別の例では、コンテキストモデル122のビット精度がKに等しいとき、上記のテーブルのエントリ、すなわち、m_MappedProb[128]は、m_MappedProb[i] = Ceil(2^K * prob[i])のように定義され得る。別の例では、m_MappedProb[i] = Ceil(2^K * prob[i] + 0.5)である。この例では、関数Ceil(x)は、xよりも大きいかまたはそれに等しい最も小さい整数(smallest integer)を示し、iは、確率インデックス、すなわち、HEVCにおいて使用されるinitStateを示す。アレイprob[128]は、HEVCによって使用されるシンボル「1」の128個の可能な確率(possible probability)を表す。

【0167】

10

20

30

【数17】

```

Prob[128] = {
  0.018750, 0.019753, 0.020810, 0.021923, 0.023096, 0.024332, 0.025633,
  0.027005, 0.028450, 0.029972, 0.031575, 0.033264, 0.035044, 0.036919,
  0.038894, 0.040975, 0.043167, 0.045476, 0.047909, 0.050473, 0.053173,
  0.056018, 0.059014, 0.062172, 0.065498, 0.069002, 0.072694, 0.076583,
  0.080680, 0.084996, 0.089543, 0.094334, 0.099381, 0.104698, 0.110299,
  0.116200, 0.122417, 0.128966, 0.135866, 0.143134, 0.150792, 0.158859,
  0.167358, 0.176312, 0.185744, 0.195682, 0.206151, 0.217180, 0.228799,
  0.241039, 0.253935, 0.267520, 0.281833, 0.296911, 0.312795, 0.329530,
  0.347159, 0.365732, 0.385299, 0.405912, 0.427629, 0.450507, 0.474609, 10
  0.500000, 0.500000, 0.525391, 0.549493, 0.572371, 0.594088, 0.614701,
  0.634268, 0.652841, 0.670470, 0.687205, 0.703089, 0.718167, 0.732480,
  0.746065, 0.758961, 0.771201, 0.782820, 0.793849, 0.804318, 0.814256,
  0.823688, 0.832642, 0.841141, 0.849208, 0.856866, 0.864134, 0.871034,
  0.877583, 0.883800, 0.889701, 0.895302, 0.900619, 0.905666, 0.910457,
  0.915004, 0.919320, 0.923417, 0.927306, 0.930998, 0.934502, 0.937828,
  0.940986, 0.943982, 0.946827, 0.949527, 0.952091, 0.954524, 0.956833,
  0.959025, 0.961106, 0.963081, 0.964956, 0.966736, 0.968425, 0.970028,
  0.971550, 0.972995, 0.974367, 0.975668, 0.976904, 0.978077, 0.979190,
  0.980247, 0.981250
}

```

20

【0168】

[0156] 図4に戻ると、いくつかの場合には、エントロピー符号化ユニット56またはビデオエンコーダ20の別のユニットは、エントロピーコーディングに加えて、他のコーディング機能を実行するように構成され得る。たとえば、エントロピー符号化ユニット56は、C UとP Uとのためのコード化パターン(C B P : coded block pattern)値を決定するように構成され得る。また、いくつかの場合には、エントロピー符号化ユニット56は係数のランレンジスコーディングを実行し得る。さらに、エントロピー符号化ユニット56、または他の処理ユニットはまた、量子化行列の値など、他のデータをコーディングし得る。

【0169】

30

[0157] 上記で説明されたように、逆量子化ユニット(inverse quantization unit)58および逆変換処理ユニット60は、たとえば、参照ブロックとして後で使用するために、ピクセル領域において残差ブロックを再構成するために、それぞれ逆量子化および逆変換を適用する。動き補償ユニット46は、残差ブロックを参照フレームメモリ64のフレームのうちの1つの予測ブロックに加算することによって参照ブロックを計算し得る。動き補償ユニット46はまた、動き推定において使用するためのサブ整数ピクセル値を計算するために、再構成された残差ブロックに1つまたは複数の補間フィルタを適用し得る。加算器62は、参照フレームメモリ64に記憶するための再構成されたビデオブロックを生成するために、動き補償ユニット46によって生成された動き補償予測ブロックに再構成された残差ブロックを加算する。再構成されたビデオブロックは、後続のビデオフレーム中のブロックをインターコーディングするために動き推定ユニット44および動き補償ユニット46によって参照ブロックとして使用され得る。

40

【0170】

[0158] 図6は、本開示で説明される技法を実装し得るビデオデコーダ30の一例を示すブロック図である。この場合も、ビデオデコーダ30は、例示のためにH E V C コーディングのコンテキストにおいて説明されるが、他のコーディング規格に関して本開示を限定するものではない。その上、ビデオデコーダ30は、範囲拡張に従って技法を実装するように構成され得る。

【0171】

[0159] 図6の例では、ビデオデコーダ30は、ビデオデータメモリ69と、エントロ

50

ピー復号ユニット70と、予測処理ユニット71と、逆量子化処理ユニット76と、逆変換処理ユニット78と、加算器80と、参照ピクチャメモリ82とを含み得る。予測処理ユニット71は、動き補償ユニット72とイントラ予測ユニット74とを含む。ビデオデコーダ30は、いくつかの例では、図4からのビデオエンコーダ20に関して説明された符号化パスに概して相反する復号パスを実行し得る。

【0172】

[0160] ビデオデータメモリ69は、ビデオデコーダ30の構成要素によって復号されるべき、符号化ビデオビットストリーム(encoded video bitstream)などのビデオデータを記憶し得る。ビデオデータメモリ69に記憶されるビデオデータは、たとえば、ストレージデバイス34から、カメラなどのローカルビデオソースから、ビデオデータのワイヤードまたはワイヤレスネットワーク通信を介して、あるいは物理データ記憶媒体にアクセスすることによって取得され得る。ビデオデータメモリ69は、符号化ビデオビットストリームからの符号化ビデオデータを記憶するコード化ピクチャバッファ(CPB:code d picture buffer)を形成し得る。

【0173】

[0161] 参照ピクチャメモリ82は、(たとえば、イントラコーディングモードまたはインターフォーマンスモードで)ビデオデコーダ30によってビデオデータを復号する際に使用するための参照ビデオデータを記憶する復号ピクチャバッファ(DPB:decoded picture buffer)の一例である。ビデオデータメモリ69および参照ピクチャメモリ82は、同期DRAM(SDRAM)を含むダイナミックランダムアクセスメモリ(DRAM)、磁気抵抗RAM(MRAM)、抵抗性RAM(RRAM)、または他のタイプのメモリデバイスなど、様々なメモリデバイスのうちのいずれかによって形成され得る。ビデオデータメモリ69および参照ピクチャメモリ82は、同じメモリデバイスまたは別個のメモリデバイスによって与えられ得る。様々な例では、ビデオデータメモリ69は、ビデオデコーダ30の他の構成要素とともにオンチップであるか、またはそれらの構成要素に対してオフチップであり得る。

【0174】

[0162] 復号プロセス中に、ビデオデコーダ30は、ビデオエンコーダ20から、符号化ビデオスライスのビデオブロックと、関連するシンタックス要素とを表す符号化ビデオビットストリームを受信する。ビデオデコーダ30のエントロピー復号ユニット70は、量子化された係数と、動きベクトルまたはイントラ予測モードインジケータと、他のシンタックス要素とを生成するために、ビットストリームをエントロピー復号(entropy decoding)する。いくつかの例では、エントロピー復号ユニット70は、エンコーダによって使用されるプロセスとは概して逆であるプロセスを適用し得る。エントロピー復号ユニット70は、変換係数の1次元アレイを取り出すために、符号化ビットストリームに対してエントロピー復号プロセスを実行する。使用されるエントロピー復号プロセスは、ビデオエンコーダ20によって使用されたエントロピーコーディング(たとえば、CABAC、CAVLC、PIPE、または上記で説明された他のプロセス)に依存する。本開示で説明される技法によれば、エントロピー復号ユニット70は、本開示で説明されるように、たとえばCABACプロセス内で、BACプロセスを適用し得る。エンコーダによって使用されたエントロピーコーディングプロセスは、符号化ビットストリーム中でシグナリングされ得るか、または所定のプロセスであり得る。

【0175】

[0163] エントロピー復号ユニット70は、動きベクトルと他のシンタックス要素とを動き補償ユニット72に転送する。ビデオデコーダ30は、ビデオスライスレベルおよび/またはビデオブロックレベルでシンタックス要素を受信し得る。

【0176】

[0164] 図7は、本開示の技法による、CABACを実行するように構成され得る例示的なエントロピー復号ユニット70のプロック図である。図7のエントロピー復号ユニット70は、図5で説明されたエントロピー符号化ユニット56の様式とは逆の様式でCA

10

20

30

40

50

B A Cを実行する。ビットストリーム218からのコード化ビットがエントロピー復号ユニット70に入力される。コード化ビットは、それらがバイパスモードを使用してエントロピーコーディングされたのか、通常モードを使用してエントロピーコーディングされたのか否かに基づいて、コンテキストモデル220またはバイパスコーディングエンジン222のいずれかに供給される。コード化ビットがバイパスモードでコーディングされた場合、バイパス復号エンジンは、たとえば、バイナリ値シンタックス要素または非バイナリシンタックス要素のビンを取り出すために、ゴロム・ライスまたは指数ゴロム復号を使用することになる。

【0177】

[0165] コード化ビットが通常モードでコーディングされた場合、コンテキストモデル220はコード化ビットのための確率モデルを決定し得、通常復号エンジン224は、非バイナリ値シンタックス要素のビン（または、バイナリ値の場合、シンタックス要素自体）を生成するためにコード化ビットを復号し得る。コンテキストおよび確率状態がコンテキストモデル220によって決定された後、通常復号エンジン224はビン値に対してB A Cを実行する。

【0178】

[0166] 本開示の1つまたは複数の技法によれば、バイナリ算術コーディングプロセスにおいて確率状態を更新するために使用される変数の同じ値（たとえば、ウィンドウサイズ、スケーリングファクタ（ ）、および確率更新速度のうちの1つまたは複数）を使用することとは対照的に、エントロピー符号化ユニット56は、異なるコンテキストおよび/または異なるシンタックス要素について変数の異なる値を使用し得る。たとえば、コンテキストモデル220は、複数のコンテキストのうちのコンテキストについて、バイナリ算術コーディングプロセスにおいて確率状態を更新するために使用される変数の値を決定し、決定された値に基づいて確率状態を更新し得る。

【0179】

[0167] いくつかの例では、次の確率状態を決定するためにコンテキストモデル220によって使用されるウィンドウサイズはコンテキスト依存にされ得る。たとえば、コンテキストモデル220は、異なるコンテキストについて異なるウィンドウサイズを使用し得る。一例として、コンテキストモデル220は、複数のコンテキストのうちの第1のコンテキストのための第1のウィンドウサイズを決定し、第1のウィンドウサイズとは異なる、複数のコンテキストのうちの第2のコンテキストのための第2のウィンドウサイズを決定し得る。

【0180】

[0168] いくつかの例では、上記のコンテキストモデル依存更新方法を、J C T V C - F 254およびJ C T V - G 764におけるものなど、カウンタベース算術コーダに組み込むとき、ウィンドウサイズの値はコンテキストに依存し得る。さらに、各コンテキストは、式(4)からの確率 P_i に加えて、ウィンドウサイズにさらに関連し得る。

【0181】

[0169] いくつかの例では、コンテキストモデル220は、 2^M に等しいことがあるウィンドウサイズ W を使用し得、ここで、 M は正の整数であり得る。したがって、いくつかのコンテキストは同じ M 値を有し得るが、各コンテキストは、他のコンテキストとは異なり得る、それ自体の M 値を有し得る。

【0182】

[0170] いくつかの例では、コンテキストモデル220は、ウィンドウサイズのあらかじめ定義されたセットからウィンドウサイズを決定し得る。たとえば、可能な M 値のセットがあらかじめ定義され得、たとえば、 M は、両端値を含む、4から7までにわたることがある。いくつかの例では、エントロピー復号ユニット70は、スライスヘッダ、あるいはピクチャパラメータセット、アクティブパラメータセット、シーケンスパラメータセット、またはビデオパラメータセットを含むパラメータセットから、可能なウィンドウサイズのセットの指示（たとえば、可能な M 値のセットの指示）を復号し得る。

10

20

30

40

50

【0183】

[0171] いくつかの例では、各コンテキストに関連するウィンドウサイズ（たとえば、Mの値）はあらかじめ定義され得る。いくつかの例では、ウィンドウサイズは、さらに、スライスタイプおよび／または（たとえば、HEVCではtemporal Idと呼ばれる）時間識別子に依存し得る。いくつかの例では、ウィンドウサイズは、さらに、ピクチャタイプ（またはNALユニットタイプ）、たとえば、ピクチャがランダムアクセスピクチャであるか否かに依存し得る。

【0184】

[0172] いくつかの例では、エントロピー復号ユニット70は、スライスヘッダ／ピクチャパラメータセット／アクティブパラメータセット／シーケンスパラメータセット中でなど、ビットストリームから、各コンテキストに関連するウィンドウサイズ（たとえば、Mの値）を復号し得る。たとえば、各コンテキストのためのデフォルトウィンドウサイズが最初にあらかじめ定義され得る。各それぞれのコンテキストについて、エントロピー復号ユニット70は、デフォルトウィンドウサイズがそれぞれのコンテキストのために使用されるかどうかを示す、それぞれのシンタックス要素（たとえば、フラグ）を復号し得る。デフォルトウィンドウサイズがそれぞれのコンテキストのために使用されない場合、エントロピー復号ユニット70は、デフォルトウィンドウサイズに基づいて、実際の使用されるウィンドウサイズを差分復号し得る。いくつかの例では、すべてのコンテキストの（すなわち、デフォルトウィンドウサイズが使用されるかどうかを示す）シンタックス要素が一緒に編成され得、エントロピー復号ユニット70は、これらのシンタックス要素を復号するためにランレンジスコーディングを利用し得る。いくつかの例では、コンテキストモデル220は、実際に使用されるウィンドウサイズとデフォルトウィンドウサイズとの間の差分をコーディングするとき、マッピングテーブルを利用し得る。たとえば、デフォルトのM値が6に等しい場合、可能なM値は、4、5、6、および7である。マッピングテーブルは次のように定義され得る。

【0185】

【表7】

実際のM値	4	5	6	7
コーディングされるべき値	0	1	-	2

10

20

30

【0186】

[0173] いくつかの例では、エントロピー復号ユニット70は、各コンテキストについて実際のウィンドウサイズとデフォルトウィンドウサイズとの間の差分を直接復号し得る。たとえば、デフォルトM値が4である場合、エントロピー復号ユニット70は、各コンテキストについてM-4の値を復号し得る。

【0187】

[0174] いくつかの例では、エントロピー復号ユニット70は、現在スライス中のコンテキストのためのすべてのウィンドウサイズが、前にコーディングされたスライス中の対応するコンテキストのための継承されたウィンドウサイズである（すなわち、前にコーディングされたスライス中の対応するコンテキストのためのウィンドウサイズに等しく設定される）かどうかを示す、第1のシンタックス要素を復号し得る。一例では、「前に復号されたスライス」は、現在スライスと同じスライスタイプ、または同じスライスタイプと量子化パラメータの両方、または同じスライスタイプと時間レイヤの両方、および／あるいは同じ初期化された量子化パラメータを有する、前にコーディングされたスライスとして定義され得る。いくつかの例では、前のスライスは、DPB中に存在するピクチャに属することが必要であり得、参照ピクチャとして現在ピクチャのために使用され得、特に、HEVCベースプラットフォームの場合のように、前のスライスは、参照ピクチャセット(RPS)中のピクチャ、さらには、RPSの以下のサブセット、すなわち、RefPicSetStCurBefore、RefPicSetStCurAfter、およびRefPicSetLtCurのうちの1つ中のピクチャに属することが必要とされ

40

50

得る。

【0188】

[0175] いくつかの例では、エントロピー復号ユニット70は、デフォルトウィンドウサイズが（たとえば、現在スライス中の）複数のコンテキストのために使用されるかどうかを示す第1のシンタックス要素を復号し得る。デフォルトウィンドウサイズが複数のコンテキストのために使用されない場合、エントロピー復号ユニット70は、コンテキストのためのウィンドウサイズを示す第2のシンタックス要素を復号し得る。たとえば、エントロピー復号ユニット70は、コンテキストのためのウィンドウサイズとデフォルトウィンドウサイズとの間の差分を示す第2のシンタックス要素を復号し得る。

【0189】

[0176] 別の例では、エントロピー復号ユニット70は、たとえば、前のスライスまたはピクチャからのコード化情報に基づいて、ウィンドウサイズを導出し得る。

【0190】

[0177] いくつかの例では、算術コーダにおいて次の確率状態または確率更新速度を決定するために使用される「ウィンドウサイズ」はシンタックス要素固有であり得る。たとえば、シンタックス要素のビンを符号化するためにコンテキストを使用するとき、コンテキストモデル220は、シンタックス要素タイプに基づいてコンテキストのためのウィンドウサイズを決定し得る。一例として、`inter_pred_idc`シンタックス要素のビンをコーディングするときにコンテキストを更新するために使用されるウィンドウサイズは、16（たとえば、 $M = 4$ ）であり得、`coeff_abs_level_greafter1 frags`シンタックス要素のビンをコーディングするときにコンテキストを更新するために使用されるウィンドウサイズは、128（たとえば、 $M = 7$ ）であり得る。

【0191】

[0178] いくつかの例では、上記の（1つまたは複数の）技法は特定のコンテキストに適用可能であり得る。すなわち、コンテキストのサブセットは、デフォルトウィンドウサイズではなく更新された「ウィンドウサイズ」を使用し得る。いくつかの例では、上記の（1つまたは複数の）技法は特定のスライスタイプに適用可能であり得る。

【0192】

[0179] ビンが通常復号エンジン224によって復号された後、逆方向バイナライザ（`reverse binarizer`）230は、ビンを非バイナリ値シンタックス要素の値に変換するために逆方向マッピング（`reverse mapping`）を実行し得る。

【0193】

[0180] 図6に戻ると、いくつかの例では、エントロピー復号ユニット70（または逆量子化ユニット76）は、ビデオエンコーダ20のエントロピー符号化ユニット56（または量子化ユニット54）によって使用された走査モードをミラーリングする走査を使用して受信値を走査し得る。係数の走査は逆量子化ユニット76において実行され得るが、走査は、例示のために、エントロピー復号ユニット70によって実行されるものとして説明される。さらに、説明しやすいように別個の機能ユニットとして示されているが、エントロピー復号ユニット70、逆量子化ユニット76、およびビデオデコーダ30の他のユニットの構造および機能は互いに高度に統合され得る。

【0194】

[0181] 逆量子化ユニット76は、ビットストリーム中で与えられ、エントロピー復号ユニット70によって復号された、量子化された変換係数を逆量子化、すなわち、量子化解除する。逆量子化プロセスは、たとえば、HEVCのまたはH.264復号規格によって定義されたいくつかの例と同様の、従来のプロセスを含み得る。逆量子化プロセスは、量子化の程度を決定し、同様に、適用されるべき逆量子化の程度を決定するための、CUについてビデオエンコーダ20によって計算される量子化パラメータ（quantization parameter）QPの使用を含み得る。逆量子化ユニット76は、係数が1次元アレイから2次元アレイに変換される前または変換された後に変換係数を逆量子化し得る。

10

20

30

40

50

【0195】

[0182] 逆変換処理ユニット78は、逆量子化された変換係数に逆変換を適用する。いくつかの例では、逆変換処理ユニット78は、ビデオエンコーダ20からのシグナリングに基づいて、あるいはブロックサイズ、コーディングモードなどの1つまたは複数のコーディング特性から変換を推論することによって、逆変換を決定し得る。いくつかの例では、逆変換処理ユニット78は、現在ブロックを含むLCUのための4分木のルートノードにおけるシグナリングされた変換に基づいて、現在ブロックに適用すべき変換を決定し得る。代替的に、変換は、LCU4分木中のリーフノードCUのためのTU4分木のルートにおいてシグナリングされ得る。いくつかの例では、逆変換処理ユニット78は、逆変換処理ユニット78が、復号されている現在ブロックの変換係数に2つまたはそれ以上の逆変換を適用する、カスケード逆変換(cascaded inverse transform)を適用し得る。

10

【0196】

[0183] さらに、逆変換処理ユニットは、本開示の上記で説明された技法に従って、変換ユニットパーティションを生成するために逆変換を適用し得る。

【0197】

[0184] イントラ予測処理ユニット74は、シグナリングされたイントラ予測モードと、現在フレームの、前に復号されたブロックからのデータとに基づいて、現在フレームの現在ブロックのための予測データを生成し得る。取り出された動き予測方向と、基準フレームインデックスと、計算された現在動きベクトル(たとえば、マージモードに従って隣接ブロックからコピーされた動きベクトル)とに基づいて、動き補償ユニットは現在部分のための動き補償ブロックを生成する。これらの動き補償ブロックは、本質的に、残差データを生成するために使用される予測ブロックを再現する。

20

【0198】

[0185] 動き補償ユニット72は、場合によっては、補間フィルタに基づく補間を実行して、動き補償ブロックを生成し得る。サブピクセル精度をもつ動き推定のために使用されるべき補間フィルタのための識別子が、シンタックス要素中に含まれ得る。動き補償ユニット72は、参照ブロックのサブ整数ピクセルのための補間値を計算するために、ビデオブロックの符号化中にビデオエンコーダ20によって使用された補間フィルタを使用し得る。動き補償ユニット72は、受信されたシンタックス情報に従って、ビデオエンコーダ20によって使用された補間フィルタを決定し、予測ブロックを生成するためにその補間フィルタを使用し得る。

30

【0199】

[0186] さらに、動き補償ユニット72およびイントラ予測処理ユニット74は、HEVCの例では、符号化ビデオシーケンスの(1つまたは複数の)フレームを符号化するために使用されたLCUのサイズを決定するために、(たとえば、4分木によって与えられる)シンタックス情報の一部を使用し得る。動き補償ユニット72およびイントラ予測処理ユニット74はまた、符号化ビデオシーケンスのフレームの各CUがどのように分割されるか(および、同様に、サブCUがどのように分割されるか)を記述する分割情報を決定するために、シンタックス情報を使用し得る。シンタックス情報はまた、各分割がどのように符号化されるかを示すモード(たとえば、イントラ予測またはインター予測、およびイントラ予測の場合はイントラ予測符号化モード)と、各インター符号化PUについての1つまたは複数の参照フレーム(および/またはそれらの参照フレームの識別子を含んでいる参照リスト)と、符号化ビデオシーケンスを復号するための他の情報とを含み得る。

40

【0200】

[0187] 加算器80は、復号ブロックを形成するために、残差ブロックを、動き補償ユニット72またはイントラ予測処理ユニット74によって生成される対応する予測ブロックと合成する。所望される場合、ブロックキネスアーティファクトを除去するために、復号ブロックをフィルタ処理するためにデブロッキングフィルタも適用され得る。復号ビデオブロックは、次いで、参照ピクチャメモリ82に記憶され、参照ピクチャメモリ82は、

50

参照ブロックを後続の動き補償に与え、また、（図1のディスプレイデバイス31などの）ディスプレイデバイス上での提示のために復号ビデオを生成する。

【0201】

[0188] 図8は、通常コーディングモードを使用する所与のビン値binValのためのバイナリ算術符号化プロセスを示す。算術符号化エンジンの内部状態は、通常通り、2つの量、すなわち、現在の間隔範囲Rおよび現在のコード間隔のベース（下側端点）Lによって特徴づけられる。しかしながら、（通常モードとバイパスモードの両方において）これらのレジスタをCABA Cエンジンに記憶するために必要とされる精度は、それぞれ9ビットおよび10ビットまで低減され得ることに留意されたい。確率状態インデックスをもつコンテキストにおいて観測される所与のバイナリ値binValおよびMPS（%2）の値の符号化は、以下のように4つの基本ステップのシーケンスにおいて実行される。

【0202】

[0189] 第1の主要なステップにおいて、現在の間隔が、所与の確率推定値に従って再分割される。この間隔再分割プロセスは、図8中の流れ図の最上のボックスに示されているように3つの基本演算を伴う。最初に、現在の間隔範囲Rが、4つのセルへの全範囲² R²の等区分を使用して、量子化された値Q(R)によって近似される。しかし、CABA Cエンジンにおいて対応する代表的な量子化された範囲値Q₀、Q₁、Q₂、およびQ₃を明示的に使用する代わりに、すなわち、以下の式（14）に従って、シフト演算とビットマスキング演算との組合せによって効率的に計算され得る、その量子化器インデックスのみによって対処される。

【0203】

【数18】

$$\rho = (R \gg 6) \& 3 \quad (14)$$

【0204】

[0190] 次いで、このインデックスおよび確率状態インデックスが、図8に示されているように、（近似）LPS関係のサブ間隔範囲R_{LPS}を決定するために、2DテーブルTabRangeLPS中のエントリとして使用される。ここで、テーブルTabRangeLPSは、8ビット精度での、0(>>1)63および03である場合の、p·Qのためのすべての64×4個のあらかじめ計算された積値（pre-computed product value）を含んでいる。

【0205】

[0191] MPSのためのデュアルサブ間隔範囲が与えられれば、所与のビン値binValに対応するサブ間隔が、符号化プロセスの第2のステップにおいて選定される。binValがMPS値に等しい場合、Lが不变であるように、下側サブ間隔が選定され（図8中の分岐の右経路）、そうでない場合、R_{LPS}に等しい範囲をもつ上側サブ間隔が選択される（図8中の左分岐）。通常算術符号化プロセスの第3のステップにおいて、確率状態の更新が、（たとえば、式（2）を使用して）上記で説明されたように実行され（図8中のグレーの影付きボックス）、最終的に、第4のステップは、Marp eによって説明されたようにレジスタLおよびRの再正規化（図8中の「RenormE」ボックス）からなる。

【0206】

[0192] 2DテーブルTabRangeLPSは以下のように定義され得る。

【0207】

10

20

30

40

【数 1 9】

```
TabRangeLPS[64][4]=  
{  
  { 128, 176, 208, 240},  
  { 128, 167, 197, 227},  
  { 128, 158, 187, 216},  
  { 123, 150, 178, 205},  
  { 116, 142, 169, 195},  
  { 111, 135, 160, 185},  
  { 105, 128, 152, 175},
```

{ 100, 122, 144, 166},	
{ 95, 116, 137, 158},	
{ 90, 110, 130, 150},	
{ 85, 104, 123, 142},	
{ 81, 99, 117, 135},	
{ 77, 94, 111, 128},	
{ 73, 89, 105, 122},	
{ 69, 85, 100, 116},	
{ 66, 80, 95, 110},	
{ 62, 76, 90, 104},	
{ 59, 72, 86, 99},	10
{ 56, 69, 81, 94},	
{ 53, 65, 77, 89},	
{ 51, 62, 73, 85},	
{ 48, 59, 69, 80},	
{ 46, 56, 66, 76},	
{ 43, 53, 63, 72},	
{ 41, 50, 59, 69},	
{ 39, 48, 56, 65},	
{ 37, 45, 54, 62},	
{ 35, 43, 51, 59},	
{ 33, 41, 48, 56},	
{ 32, 39, 46, 53},	20
{ 30, 37, 43, 50},	
{ 29, 35, 41, 48},	
{ 27, 33, 39, 45},	
{ 26, 31, 37, 43},	
{ 24, 30, 35, 41},	
{ 23, 28, 33, 39},	
{ 22, 27, 32, 37},	
{ 21, 26, 30, 35},	
{ 20, 24, 29, 33},	
{ 19, 23, 27, 31},	
{ 18, 22, 26, 30},	
{ 17, 21, 25, 28},	30
{ 16, 20, 23, 27},	
{ 15, 19, 22, 25},	
{ 14, 18, 21, 24},	
{ 14, 17, 20, 23},	
{ 13, 16, 19, 22},	
{ 12, 15, 18, 21},	
{ 12, 14, 17, 20},	
{ 11, 14, 16, 19},	
{ 11, 13, 15, 18},	
{ 10, 12, 15, 17},	
{ 10, 12, 14, 16},	40
{ 9, 11, 13, 15},	
{ 9, 11, 12, 14},	
{ 8, 10, 12, 14},	
{ 8, 9, 11, 13},	

```

{ 7, 9, 11, 12},
{ 7, 9, 10, 12},
{ 7, 8, 10, 11},
{ 6, 8, 9, 11},
{ 6, 7, 9, 10},
{ 6, 7, 8, 9},
{ 2, 2, 2, 2}
};

```

【0208】

[0193] 例示的なCABAC復号プロセスは、HEVC規格のセクション9.3.4.3.2.2において見つけられる。 10

【0209】

[0194] 図9は、残差4分木に基づく変換方式を示す概念図である。残差ブロックの様々な特性を適応させるために、HEVCでは残差4分木(RQT)を使用する変換コーディング構造が適用され、これは、

<http://www.hhi.fraunhofer.de/departments/video-coding-analytics/research-groups/image-video-coding/hevc-high-efficiency-video-coding/transform-coding-using-the-residual-quadtree-rqt.html>

に手短に記載されている。

【0210】

20

[0195] 各ピクチャが、特定のタイルまたはスライスについてラスタ走査順序でコーディングされるコーディングツリーユニット(CTU: coding tree unit)に分割される。CTUは、正方形ブロックであり、4分木、すなわち、コーディングツリーのルートを表す。CTUサイズは 8×8 から 64×64 ルーマサンプルにわたり得るが、一般に 64×64 が使用される。各CTUは、さらに、コーディングユニット(CU)と呼ばれるより小さい正方形ブロックにスプリットされ得る。CTUがCUに再帰的にスプリットされた後、各CUはPUおよびTUにさらに分割される。TUへのCUの区分は、4分木手法に基づいて再帰的に行われ、したがって、各CUの残差信号は、ツリー構造、すなわち、残差4分木(RQT)によってコーディングされる。RQTは、 4×4 から 32×32 ルーマサンプルまでのTUサイズを可能にする。図9は、CUが、文字a~jで標示された10個のTUを含む一例と、対応するブロック区分とを示す。RQTの各ノードは、実際は変換ユニット(TU)である。個々のTUは、深度優先トラバーサル(depth-first traversal)による再帰的Z走査に従う、アルファベット順として図に示された深度優先ツリートラバーサル順序で処理される。4分木手法は、残差信号の変動する空間周波数特性に対する変換の適応を可能にする。一般に、より大きい空間サポートを有するより大きい変換ブロックサイズは、より良い周波数解像度を与える。しかしながら、より小さい空間サポートを有するより小さい変換ブロックサイズは、より良い空間解像度を与える。その2つ、すなわち、空間解像度と周波数解像度との間のトレードオフは、たとえばレートひずみ最適化技法に基づいて、エンコーダモード決定によって選定される。レートひずみ最適化技法は、各コーディングモード(たとえば、特定のRQTスプリッティング構造)についてコーディングビットと再構成ひずみとの加重和、すなわち、レートひずみコストを計算し、最小レートひずみコストをもつコーディングモードを最良のモードとして選択する。

30

【0211】

40

[0196] 3つのパラメータ、すなわち、ツリーの最大深度(maximum depth of the tree)、最小許容変換サイズ(minimum allowed transform size)および最大許容変換サイズ(maximum allowed transform size)がRQTにおいて定義される。HEVCのいくつかの例では、最小および最大変換サイズは、前の段落で述べられたサポートされるブロック変換に対応する、 4×4 から 32×32 サンプルまでの範囲内で変動することがある。RQTの最大許容深度はTUの数を制限する。0に等しい最大深度は、各含まれたTUが

50

最大許容変換サイズ、たとえば、 32×32 に達した場合、CTUがこれ以上スプリットされ得ないことを意味する。

【0212】

[0197] すべてのこれらのパラメータは、相互作用し、RQT構造に影響を及ぼす。ルートCTUサイズが 64×64 であり、最大深度が0に等しく、最大変換サイズが 32×32 に等しい場合について考える。この場合、CTUは、さもなければ、それが、許容されない 64×64 TUにつながることになるので、少なくとも1回区分されなければならない。RQTパラメータ、すなわち、最大RQT深度、最小および最大変換サイズは、シーケンスパラメータセットレベルにおいてビットストリーム中で送信される。RQT深度に関して、イントラコード化CUとインターフォーマットコード化CUとについて異なる値が指定され、シグナリングされ得る。 10

【0213】

[0198] 4分木変換は、イントラ残差ブロックとインター残差ブロックの両方のために適用される。一般に、現在の残差4分木パーティションの同じサイズのDCT-II変換が残差ブロックのために適用される。しかしながら、現在の残差4分木ブロックが 4×4 であり、イントラ予測によって生成される場合、上記の 4×4 DST-VII変換が適用される。

【0214】

[0199] HEVCでは、より大きいサイズの変換、たとえば、 64×64 変換は、主に、および比較的より小さい解像度のビデオに対する比較的高い複雑さを考慮して、それらの限られた利益により、採用されない。 20

【0215】

[0200] 図10は、係数グループに基づく例示的な係数走査を示す概念図である。TUサイズかかわらず、変換ユニットの残差は、各々がTUの 4×4 ブロックの係数を含んでいる非重複係数グループ(CG: coefficient group)でコーディングされる。たとえば、 32×32 TUは全体として64個のCGを有し、 16×16 TUは全体として16個のCGを有する。TU内のCGは、あるあらかじめ定義された走査順序に従ってコーディングされ得る。各CGをコーディングするとき、現在CG内の係数は、 4×4 ブロックについて、あるあらかじめ定義された走査順序に従って走査され、コーディングされる。図10は、4つのCGを含んでいる 8×8 TUについての係数走査を示す。 30

【0216】

[0201] シンタックス要素テーブルは以下のように定義される。

【0217】

【表 8】

7.3.8.11 残差コーディングシンタックス

residual_coding(x0, y0, log2TrafoSize, cIdx) {		記述子
if(transform_skip_enabled_flag && !cu_transquant_bypass_flag && (log2TrafoSize == 2))	transform_skip_flag[x0][y0][cIdx]	ae(v)
last_sig_coeff_x_prefix		ae(v)
last_sig_coeff_y_prefix		ae(v)
if(last_sig_coeff_x_prefix > 3)		
last_sig_coeff_x_suffix		ae(v)
if(last_sig_coeff_y_prefix > 3)		
last_sig_coeff_y_suffix		ae(v)
lastScanPos = 16		
lastSubBlock = (1 << (log2TrafoSize - 2)) * (1 << (log2TrafoSize - 2)) - 1		
do {		
if(lastScanPos == 0) {		
lastScanPos = 16		
lastSubBlock--		
}		
lastScanPos--		
xS = ScanOrder[log2TrafoSize - 2][scanIdx][lastSubBlock][0]		
yS = ScanOrder[log2TrafoSize - 2][scanIdx][lastSubBlock][1]		
xC = (xS << 2) + ScanOrder[2][scanIdx][lastScanPos][0]		
yC = (yS << 2) + ScanOrder[2][scanIdx][lastScanPos][1]		
} while((xC != LastSignificantCoeffX) (yC != LastSignificantCoeffY))		
for(i = lastSubBlock; i >= 0; i--) {		
xS = ScanOrder[log2TrafoSize - 2][scanIdx][i][0]		
yS = ScanOrder[log2TrafoSize - 2][scanIdx][i][1]		
inferSbDcSigCoeffFlag = 0		
if((i < lastSubBlock) && (i > 0)) {		
coded_sub_block_flag[xS][yS]		ae(v)
inferSbDcSigCoeffFlag = 1		
}		
for(n = (i == lastSubBlock) ? lastScanPos - 1 : 15; n >= 0; n--) {		
xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]		
yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]		
if(coded_sub_block_flag[xS][yS] && (n > 0 !inferSbDcSigCoeffFlag)) {		
sig_coeff_flag[xC][yC]		ae(v)
if(sig_coeff_flag[xC][yC])		
inferSbDcSigCoeffFlag = 0		

10

20

30

40

```

    }

}

firstSigScanPos = 16
lastSigScanPos = -1
numGreater1Flag = 0
lastGreater1ScanPos = -1
for( n = 15; n >= 0; n-- ) {
    xC = ( xS << 2 ) + ScanOrder[ 2 ][ scanIdx ][ n ][ 0 ]
    yC = ( yS << 2 ) + ScanOrder[ 2 ][ scanIdx ][ n ][ 1 ]
    if( sig_coeff_flag[ xC ][ yC ] ) {
        if( numGreater1Flag < 8 ) {
            coeff_abs_level_greater1_flag[ n ] ae(v)
            numGreater1Flag++
            if( coeff_abs_level_greater1_flag[ n ] && lastGreater1ScanPos
                == -1 )
                lastGreater1ScanPos = n
        }
        if( lastSigScanPos == -1 )
            lastSigScanPos = n
        firstSigScanPos = n
    }
}
signHidden = ( lastSigScanPos - firstSigScanPos > 3 &&
!cu_transquant_bypass_flag )
if( lastGreater1ScanPos != -1 )
    coeff_abs_level_greater2_flag[ lastGreater1ScanPos ] ae(v)
for( n = 15; n >= 0; n-- ) {
    xC = ( xS << 2 ) + ScanOrder[ 2 ][ scanIdx ][ n ][ 0 ]
    yC = ( yS << 2 ) + ScanOrder[ 2 ][ scanIdx ][ n ][ 1 ]
    if( sig_coeff_flag[ xC ][ yC ] &&
        ( !sign_data_hiding_enabled_flag || !signHidden || ( n != firstSigScanPos ) ) )
        coeff_sign_flag[ n ] ae(v)
}
numSigCoeff = 0
sumAbsLevel = 0
for( n = 15; n >= 0; n-- ) {
    xC = ( xS << 2 ) + ScanOrder[ 2 ][ scanIdx ][ n ][ 0 ]
    yC = ( yS << 2 ) + ScanOrder[ 2 ][ scanIdx ][ n ][ 1 ]
    if( sig_coeff_flag[ xC ][ yC ] ) {
        baseLevel = 1 + coeff_abs_level_greater1_flag[ n ] +
            coeff_abs_level_greater2_flag[ n ]
    }
}

```

10

20

30

40

```

if( baseLevel == ( ( numSigCoeff < 8 ) ?
                    ( ( n == lastGreater1ScanPos ) ? 3 : 2 ) : 1 ) )
    coeff_abs_level_remaining[ n ] = ae(v)

TransCoeffLevel[ x0 ][ y0 ][ cIdx ][ xC ][ yC ] =
    ( coeff_abs_level_remaining[ n ] + baseLevel ) * ( 1 - 2 *
coeff_sign_flag[ n ] )

if( sign_data_hiding_enabled_flag && signHidden ) {
    sumAbsLevel += ( coeff_abs_level_remaining[ n ] + baseLevel )
    if( ( n == firstSigScanPos ) && ( ( sumAbsLevel % 2 ) == 1 ) )
        TransCoeffLevel[ x0 ][ y0 ][ cIdx ][ xC ][ yC ] =
            -TransCoeffLevel[ x0 ][ y0 ][ cIdx ][ xC ][ yC ]
}
numSigCoeff++
}
}
}
}

```

〔 0 2 1 8 〕

[0202] 各色成分について、現在 T U が少なくとも 1 つの非 0 係数を有するかどうかを示すために、1 つのフラグが最初にシグナリングされ得る。少なくとも 1 つの非 0 係数がある場合、T U 中の係数走査順序 (coefficient scan order) における最後有意係数 (last significant coefficient) の位置が、変換ユニットの左上隅に対する座標を用いて明示的にコーディングされる。座標の垂直または水平成分は、そのプレフィックスおよびサフィックスによって表され、ここにおいて、プレフィックスは短縮ライス (T R : truncated rice) を用いて 2 値化され、サフィックスは固定長を用いて 2 値化される。

〔 0 2 1 9 〕

〔0203〕 セマンティクス：

[0204] `last_sig_coeff_x_prefix` は、変換ブロック内の走査順序における最後有意係数の列位置のプレフィックスを指定する。`last_sig_coeff_ff_x_prefix` の値は、両端値を含む、 $0 \sim (\log_2 \text{TrafoSize} < < 1) - 1$ の範囲内にあるものとする。

〔 0 2 2 0 〕

[0205] `last_sig_coeff_y_prefix` は、変換ブロック内の走査順序における最後有意係数の行位置のプレフィックスを指定する。`last_sig_coeff_ff_y_prefix` の値は、両端値を含む、 $0 \sim (\log_2 \text{TrafoSize} \leq 1) - 1$ の範囲内にあるものとする。

[0 2 2 1]

[0206] `last_sig_coeff_x_suffix` は、変換ブロック内の走査順序における最後有意係数の列位置のサフィックスを指定する。`last_sig_coeff_ff_x_suffix` の値は、両端値を含む、`0 ~ (1 << ((last_sig_coeff_ff_x_prefix >> 1) - 1)) - 1` の範囲内にあるものとする。

(02221)

[0207] 変換ブロック Last Significant Coeff X内の走査順序における最後有意係数の列位置は、以下のように導出される。

last_sigmoid_coeff_x_suffix が存在しない場合、以下が適用される。

(0 2 2 3)

〔数20〕

LastSignificantCoeffX = last sig coeff x prefix

【0224】

そうでない場合 (last_sig_coeff_ff_x_sufffix が存在する)、以下が適用される。

【0225】

【数21】

```
LastSignificantCoeffX = (1 << ((last_sig_coeff_x_prefix >> 1) - 1)) *
(2 + (last_sig_coeff_x_prefix & 1)) + last_sig_coeff_x_suffix
```

【0226】

[0208] last_sig_coeff_y_sufffix は、変換ブロック内の走査順序における最後有意係数の行位置のサフィックスを指定する。last_sig_coeff_y_sufffix の値は、両端値を含む、 $0 \sim (1 << ((\text{last_sig_coeff_y_prefix} >> 1) - 1)) - 1$ の範囲内にあるものとする。

10

【0227】

[0209] 変換ブロック LastSignificantCoeffY 内の走査順序における最後有意係数の行位置は、以下のように導出される。

last_sig_coeff_y_sufffix が存在しない場合、以下が適用される。

【0228】

【数22】

```
LastSignificantCoeffY = last_sig_coeff_y_prefix
```

20

【0229】

そうでない場合 (last_sig_coeff_y_sufffix が存在する)、以下が適用される。

【0230】

【数23】

```
LastSignificantCoeffY = (1 << ((last_sig_coeff_y_prefix >> 1) - 1)) *
(2 + (last_sig_coeff_y_prefix & 1)) + last_sig_coeff_y_suffix
```

【0231】

[0210] scanIdx が 2 に等しいとき、座標は以下のようにスワップされる。

【0232】

30

【数24】

```
(LastSignificantCoeffX, LastSignificantCoeffY) = Swap(LastSignificantCoeffX,
LastSignificantCoeffY)
```

【0233】

[0211] コーディングされたそのような位置、また CG の係数走査順序とともに、(走査順序における) 最後 CG を除く CG のために、それが非 0 係数を含んでいるかどうかを示す 1 つのフラグがさらにシグナリングされる。

【0234】

[0212] CG フラグのコンテキストモデリング。1 つの CG が非 0 係数を有するかどうか、すなわち、CG フラグ (HEVC 仕様における code_subblock_flag) をコーディングするとき、隣接 CG の情報が、コンテキストを構築するために利用される。より具体的に言えば、CG フラグをコーディングするためのコンテキスト選択は次のように定義される。

40

【0235】

【数25】

(右 CG 利用可能 && 右 CG のフラグが 1 に等しい) || (下 CG 利用可能 && 下 CG のフラグが 1 に等しい)

【0236】

[0213] ここで、右および下 CG は、現在 CG に近接する 2 つの隣接 CG である。たとえば、図 10 では、左上 4 × 4 ブロックをコーディングするとき、右 CG は右上 4 × 4 ブ

50

ロックとして定義され、下 C G は左下 4×4 ブロックとして定義される。

【0237】

[0214] クロマおよびルーマは、コンテキストの異なるセットを使用するが、それらのうちの 1 つを選択するための同じルールを用いることに留意されたい。

【0238】

[0215] コンテキストインデックス増分の導出の詳細は、HEVC の 9.3.4.2.4 において見つけられ得る。

【0239】

[0216] 1 つの C G 内での変換係数コーディング。非 0 係数を含んでいることがある C G について、有意フラグ (significant_flag)、(coeff_abs_level_greater1_flag、coeff_abs_level_greater2_flag および coeff_abs_level_remaining) を含む) 係数の絶対値および符号情報 (coeff_sign_flag) が、あらかじめ定義された 4×4 係数走査順序に従って各係数についてさらにコーディングされ得る。変換係数レベルのコーディングは複数の走査パスに分離される。
10

【0240】

[0217] 1) 第 1 の binコーディングの第 1 のパス。このパスにおいて、特定の変換係数が 0 に等しいことが導出され得ることを除いて、1 つの C G 内の各位置における変換係数のすべての第 1 の bin (または bin インデックス 0、bin0) がコーディングされる。
20

【0241】

[0218] 変数 sig_Ctx は、現在 TU の左上 position に対する現在ロケーションと、色成分インデックス cIdx と、変換ブロックサイズと、シンタックス要素 coded_subblock_flag の前に復号された bin とに依存する。異なるルールが TU サイズに応じて適用される。コンテキストインデックス増分の選択の例示的な詳細は、HEVC の 9.3.4.2.5 において定義されている。

【0242】

[0219] 2) 第 2 の binコーディングの第 2 のパス。coeff_abs_level_greater1_flag のコーディングがこのパスにおいて適用される。コンテキストモデリングは、色成分インデックスと、現在サブブロック走査インデックスと、現在サブブロック内の現在係数走査インデックスとに依存する。コンテキストインデックス増分の選択の例示的な詳細は、HEVC の 9.3.4.2.6 において定義されている。
30

【0243】

[0220] 3) 第 3 の binコーディングの第 3 のパス。coeff_abs_level_greater2_flag のコーディングがこのパスにおいて適用される。コンテキストモデリングは、coeff_abs_level_greater1_flag によって使用されるものと同様である。コンテキストインデックス増分の選択の例示的な詳細は、HEVC の 9.3.4.2.7 において定義されている。

【0244】

[0221] スループットを改善するために、第 2 および第 3 のパスは C G 中のすべての係数を処理するとは限らないことに留意されたい。C G 中の最初の 8 つの coeff_abs_level_greater1_flag が通常モードでコーディングされる。その後、値は、シンタックス coeff_abs_level_remaining によって第 5 のパスにおいてバイパスモードでコーディングされるために残される。同様に、1 よりも大きい値をもつ C G 中の第 1 の係数のための coeff_abs_level_greater2_flag のみがコーディングされる。C G の 1 よりも大きい値をもつ係数の残りは、値をコーディングするために coeff_abs_level_remaining を使用する。この方法は、係数レベルのための通常 bin の数を、C G ごとに最大 9 つ、すなわち、coeff_abs_level_greater1_flag について 8 つおよび coeff_abs_level_greater2_flag について 1 つ
40
50

に制限する。

【0245】

[0222] 4) 符号情報の第4のパス。HEVCのいくつかの例では、各非0係数の符号は、バイパスモードで第4の走査パスにおいてコーディングされる。各CGについて、および基準に応じて、(逆方向走査順序における)最後非0係数(last nonzero coefficient)の符号を符号化することは、符号データヒッディング(SDH:sign data hidding)を使用するとき、単に省略される。代わりに、符号値は、偶数が「+」に対応し、奇数が「-」に対応するという、あらかじめ定義された規約を使用してCGのレベルの和のパリティ中に埋め込まれる。SDHを使用するための基準は、CGの第1の非0係数と最後非0係数との間の走査順序における距離である。この距離が4に等しいかまたはそれよりも大きい場合、SDHが使用される。4のこの値は、それがHEVCテストシーケンスに対して最も大きい利得を与えるので選定された。10

【0246】

[0223] 5) 残りのビンの最後のパス。残りのビンが、さらなる走査パスにおいてコーディングされる。係数のbaseLevelが次のように定義されるとする。

【0247】

【数26】

```
baseLevel = significant_flag + coeff_abs_level_greater1_flag+
coeff_abs_level_greater2_flag
```

【0248】

[0224] ここで、フラグは、0または1の値を有し、存在しない場合、0であると推論される。その場合、係数の絶対値は単に以下の通りである。20

【0249】

【数27】

```
absCoeffLevel = baseLevel + coeff_abs_level_remaining
```

【0250】

[0225] ライスパラメータ(Rice parameter)は、各CGの最初に0に設定され、それは、以下のようにパラメータの前の値と現在の絶対レベルとに応じて条件付きで更新される。

【0251】

【数28】

```
if absCoeffLevel > 3 × 2m, m= min(4,m + 1)
```

【0252】

[0226] シンタックス要素coeff_abs_level_remainingはバイパスモードでコーディングされ得る。さらに、HEVCのいくつかの例は、小さい値についてはゴロムライスコード(Golomb-Rice code)を採用し、より大きい値については指数ゴロムコード(Exp-Golomb code)に切り替わる。コード間の遷移点は、一般に、単項コード長が4に等しいときである。パラメータ更新プロセスは、2値化が、分布において大きい値が観測されたとき、係数統計値に適応することを可能にする。

【0253】

[0227] integer_pred_idcのコンテキストモデリング。integer_pred_idcは、現在予測ユニットのためにlist0が使用されるのか、list1が使用されるのか、双予測が使用されるのかを指定する。シンタックス要素は、その両方がCABACコンテキストコーディングされる、最高2つのビンを有する。2値化されたビンストリングは以下のように定義される。

【0254】

【表9】

inter_pred_idc の値	ビンストリング	ビンストリング
	$(nPbW + nPbH) \neq 12$	$(nPbW + nPbH) \neq 12$
0	00	00
1	01	01
2	1	1

【0255】

[0228] ここで、nPbWおよびnPbHは、それぞれ、現在ルーマ予測プロック幅および高さを表す。

10

【0256】

[0229] 各インターフェース化スライス、たとえば、PスライスまたはBスライスについて、コンテキスト選択は以下のルールに基づく。

【0257】

- $(nPbW + nPbH)$ が 12 に等しくない場合、第 1 のビンは、4 つのコンテキストを使用してコーディングされ、第 2 のビンは、1 つのコンテキストを用いてコーディングされる。第 1 のビンのコンテキスト選択は現在 CU 深度従う。HEVC では、CU 深度は、両端値を含む、0 ~ 3 の範囲内にある。

【0258】

[0230] 図 11 は、本開示の 1 つまたは複数の技法による、異なるウィンドウサイズでコンテキストベースエントロピー符号化を実行するための例示的なプロセスを示すフローチャートである。図 11 の技法は、図 1 および図 4 に示されたビデオエンコーダ 20 など、ビデオエンコーダによって実行され得る。説明の目的で、図 11 の技法は、図 1 および図 4 のビデオエンコーダ 20 のコンテキスト内で説明されるが、ビデオエンコーダ 20 の構成とは異なる構成を有するビデオエンコーダが図 11 の技法を実行し得る。

20

【0259】

[0231] ビデオエンコーダ 20 は、コンテキストベースエントロピーコーディング（たとえば、CABAC）を使用して符号化されるべきビンストリング（たとえば、1 次元バイナリベクトル）を取得する（1102）。たとえば、ビデオエンコーダ 20 のエントロピー符号化ユニット 56 は、ビデオエンコーダ 20 の予測処理ユニット 42 から受信されたシンタックス要素を 2 値化することによってビンストリングを取得し得る。

30

【0260】

[0232] 本開示の 1 つまたは複数の技法によれば、ビデオエンコーダ 20 は、複数のコンテキストのうちのコンテキストのためのあらかじめ定義された初期化値を取得する（1104）。たとえば、ビデオエンコーダ 20 のエントロピー符号化ユニット 56 は、N ビット精度でのストレージからあらかじめ定義された初期化値を取得し得る。

【0261】

[0233] ビデオエンコーダ 20 は、あらかじめ定義された初期化値に基づいて、ビデオデータのスライスのためのコンテキストの初期確率状態を決定する（1106）。いくつかの例では、ビデオエンコーダ 20 は、初期確率状態を決定するためにルックアップテーブルを使用し得る。たとえば、エントロピー符号化ユニット 56 は、あらかじめ定義された初期化値に基づいて、傾斜値とオフセット値とを決定し得る。エントロピー符号化ユニット 56 は、傾斜値と、オフセット値と、ビデオデータのスライスの量子化パラメータとに基づいて中間値を決定し得る。エントロピー符号化ユニット 56 は、ルックアップテーブルを使用して、中間値を初期確率状態にマッピングし得る。いくつかの例では、コンテキストのための可能な確率状態の数は、2 の N 乗よりも大きい（すなわち、 2^N よりも大きい）。

40

【0262】

[0234] ビデオエンコーダ 20 は、ビデオビットストリーム中およびコンテキストの初期確率状態に基づいて、ビンストリングのビンを符号化する（1108）。たとえば、

50

エントロピー符号化ユニット 56 は、コンテキストの最終コード化確率間隔内の確率に対する値またはポインタを表すバイナリストリームを出力し得る。

【 0 2 6 3 】

[0235] 図 12 は、本開示の 1 つまたは複数の技法による、異なるウィンドウサイズでコンテキストベースエントロピー復号を実行するための例示的なプロセスを示すフローチャートである。図 12 の技法は、図 1 および図 6 に示されたビデオデコーダ 30 など、ビデオデコーダによって実行され得る。説明の目的で、図 12 の技法は、図 1 および図 6 のビデオデコーダ 30 のコンテキスト内で説明されるが、ビデオデコーダ 30 の構成とは異なる構成を有するビデオデコーダが図 12 の技法を実行し得る。

【 0 2 6 4 】

[0236] ビデオデコーダ 30 は、ビデオビットストリームから、コンテキストベースエントロピーコーディングを使用して復号されるべきビンストリング（たとえば、1 次元バイナリベクトル）を取得する（1202）。たとえば、ビデオデコーダ 30 のエントロピー復号ユニット 70 は、ビデオデータメモリ 69 からビンストリングを取得し得る。いくつかの例では、ビンストリングは、コンテキストの最終コード化確率間隔内の確率に対する値またはポインタを表し得る。いくつかの例では、コンテキストベースエントロピーコーディングはコンテキスト適応型バイナリ算術コーディング（C A B A C）を備え得る。

【 0 2 6 5 】

[0237] 本開示の 1 つまたは複数の技法によれば、ビデオデコーダ 30 は、複数のコンテキストのうちのコンテキストのためのあらかじめ定義された初期化値を取得する（1204）。たとえば、ビデオデコーダ 30 のエントロピー復号ユニット 70 は、N ビット精度でのストレージからあらかじめ定義された初期化値を取得し得る。

【 0 2 6 6 】

[0238] ビデオデコーダ 30 は、あらかじめ定義された初期化値に基づいて、ビデオデータのスライスのためのコンテキストの初期確率状態を決定する（1206）。いくつかの例では、ビデオデコーダ 30 は、初期確率状態を決定するためにロックアップテーブルを使用し得る。たとえば、エントロピー復号ユニット 70 は、あらかじめ定義された初期化値に基づいて、傾斜値とオフセット値とを決定し得る。エントロピー復号ユニット 70 は、傾斜値と、オフセット値と、ビデオデータのスライスの量子化パラメータとに基づいて中間値を決定し得る。エントロピー復号ユニット 70 は、ロックアップテーブルを使用して、中間値を初期確率状態にマッピングし得る。いくつかの例では、コンテキストのための可能な確率状態の数は、2 の N 乗よりも大きい（すなわち、 2^N よりも大きい）。

【 0 2 6 7 】

[0239] ビデオデコーダ 30 は、コンテキストの初期確率状態に基づいて、ビンストリングのビンを復号する（1208）。ビデオデコーダ 30 は、復号されたビンとコンテキストの初期確率状態に基づいて、コンテキストの更新された確率状態を決定し得る。ビデオデコーダ 30 は、コンテキストの更新された確率状態に基づいて、別のビンを復号する（1206）。

【 0 2 6 8 】

[0240] 以下の番号付けされた例は、本開示の 1 つまたは複数の態様を示し得る。

【 0 2 6 9 】

[0241] 例 1. ビデオデータをエントロピーコーディングする方法であって、本方法は、ビデオデータのスライス中のシンタックス要素のための値をエントロピーコーディングするために、コンテキスト適応型エントロピーコーディングプロセスにおいて使用される複数のコンテキストのうちのコンテキストのためのあらかじめ定義された初期化値を取得することと、ここにおいて、あらかじめ定義された初期化値が N ビット精度で記憶される、あらかじめ定義された初期化値に基づいて、ビデオデータのスライスのためのコンテキストの初期確率状態を決定することと、ここにおいて、コンテキストのための可能な確率状態の数が 2 の N 乗よりも大きい、コンテキストの初期確率状態に基づいて、シンタックス要素のための値のビンをエントロピーコーディングすることとを備える、方法。

10

20

30

40

50

【0270】

[0242] 例 2 . あらかじめ定義された初期化値に基づいて、傾斜値とオフセット値とを決定することをさらに備え、ここにおいて、初期確率状態を決定することが、傾斜値と、オフセット値と、ビデオデータのスライスの量子化パラメータとに基づいて中間値を決定することと、ルックアップテーブルを使用して、中間値を初期確率状態にマッピングすることとを備える、例 1 に記載の方法。

【0271】

[0243] 例 3 . ルックアップテーブルが、2 の N 乗よりも小さいかまたはそれに等しい数のエントリを含む、例 2 に記載の方法。

【0272】

[0244] 例 4 . エントリの数が中間値の可能な値の数 (a number of possible values) に等しい、例 3 に記載の方法。

【0273】

[0245] 例 5 . コンテキストの初期確率状態を決定することが、ルックアップテーブルを使用してコンテキストの初期確率状態を決定することを備え、ここにおいて、ルックアップテーブル中の値が以下の式、 $\text{M a p p e d P r o b}[i] = \text{C e i l}(2^M * p r o b[i] + \text{オフセット})$ に従って定義され、上式で、 $\text{M a p p e d P r o b}[i]$ は、ルックアップテーブル中の i 番目の値であり、 $prob[i]$ は、1 シンボルの可能な確率のセットを表すテーブル中の i 番目の値を表し、2 の M 乗は、コンテキストのための可能な確率状態の数を表し、 $\text{C e i l}(x)$ は、 x よりも大きいかまたはそれに等しい最も小さい整数を示す関数である、例 1 から 4 の任意の組合せに記載の方法。

【0274】

[0246] 例 6 . $prob[i]$ が 1 シンボルの i 番目の可能な確率を表す、例 1 から 5 の任意の組合せに記載の方法。

【0275】

[0247] 例 7 . コンテキストのための可能な確率状態の数が、2 の N 乗よりも大きいかまたはそれに等しい、例 1 から 6 の任意の組合せに記載の方法。

【0276】

[0248] 例 8 . N が 8 であり、コンテキストのための可能な確率状態の数が 2 の 15 乗である、例 1 から 6 の任意の組合せに記載の方法。

【0277】

[0249] 例 9 . オフセットが 0 . 5 または 0 に等しい、例 1 から 8 の任意の組合せに記載の方法。

【0278】

[0250] 例 10 . あらかじめ定義された初期化値に基づいて、傾斜値とオフセット値とを決定することをさらに備え、ここにおいて、初期確率状態を決定することが、傾斜値と、オフセット値と、ビデオデータのスライスの量子化パラメータとに基づいて中間値を決定することと、中間値と初期確率状態との間のマッピング関数 (mapping function) を使用して、中間値に基づいて初期確率状態を決定することとを備える、例 1 から 9 の任意の組合せに記載の方法。

【0279】

[0251] 例 11 . エントロピーコーディングすることが、ビンをエントロピー符号化することを備える、例 1 から 10 の任意の組合せに記載の方法。

【0280】

[0252] 例 12 . エントロピーコーディングすることが、ビンをエントロピー復号することを備える、例 1 から 10 の任意の組合せに記載の方法。

【0281】

[0253] 例 13 . コンテキスト適応型エントロピーコーディングプロセスが、コンテキスト適応型バイナリ算術コーディング (C A B A C) プロセス、またはコンテキスト適応型可変長コーディング (C A V L C) プロセスを備える、例 1 から 12 の任意の組合せに

10

20

30

40

50

記載の方法。

【0282】

[0254] 例14. ビデオデータのエントロピーコーディングのための装置であって、本装置が、シンタックス要素のための値をエントロピーコーディングするために、コンテキスト適応型エントロピーコーディングプロセスにおいて使用される複数のコンテキストを記憶するように構成されたメモリと、例1から13の任意の組合せに記載の方法を実行するように構成された1つまたは複数のプロセッサとを備える、装置。

【0283】

[0255] 例15. 本装置が、集積回路、マイクロプロセッサ、またはワイヤレス通信デバイス (wireless communication device) のうちの少なくとも1つを備える、例14に記載の装置。

10

【0284】

[0256] 例16. 優号ビデオデータを表示するように構成されたディスプレイ (display) をさらに備える、例14から15の任意の組合せに記載の装置。

【0285】

[0257] 例17. ビデオデータをキャプチャするように構成されたカメラ (camera) をさらに備える、例14から16の任意の組合せに記載の装置。

【0286】

[0258] 例18. ビデオデータのエントロピーコーディングのための装置であって、本装置が、例1から13の任意の組合せに記載の方法を実行するための手段を備える、装置。

20

【0287】

[0259] 例19. 実行されたとき、ビデオコーディングデバイスの1つまたは複数のプロセッサに、例1から13の任意の組合せに記載の方法を実行させる命令を記憶するコンピュータ可読記憶媒体。

【0288】

[0260] 1つまたは複数の例では、説明された機能は、ハードウェア、ソフトウェア、ファームウェア、またはそれらの任意の組合せで実装され得る。ソフトウェアで実装される場合、機能は、1つまたは複数の命令またはコードとしてコンピュータ可読媒体上に記憶されるか、あるいはコンピュータ可読媒体を介して送信され、ハードウェアベースの処理ユニットによって実行され得る。コンピュータ可読媒体は、データ記憶媒体などの有形媒体に対応する、コンピュータ可読記憶媒体を含み得るか、または、たとえば、通信プロトコルに従って、ある場所から別の場所へのコンピュータプログラムの転送を可能にする任意の媒体を含む通信媒体を含み得る。このようにして、コンピュータ可読媒体は、概して、(1)非一時的である有形コンピュータ可読記憶媒体、あるいは(2)信号または搬送波などの通信媒体に対応し得る。データ記憶媒体は、本開示で説明された技法の実装のための命令、コードおよび/またはデータ構造を取り出すために、1つまたは複数のコンピュータあるいは1つまたは複数のプロセッサによってアクセスされ得る、任意の利用可能な媒体であり得る。コンピュータプログラム製品はコンピュータ可読媒体を含み得る。

30

【0289】

[0261] 限定ではなく例として、そのようなコンピュータ可読記憶媒体は、RAM、ROM、EEPROM (登録商標)、CD-ROMまたは他の光ディスクストレージ、磁気ディスクストレージ、または他の磁気ストレージデバイス、フラッシュメモリ、あるいは命令またはデータ構造の形態の所望のプログラムコードを記憶するために使用され得、コンピュータによってアクセスされ得る、任意の他の媒体を備えることができる。また、いかなる接続もコンピュータ可読媒体と適切に呼ばれる。たとえば、命令が、同軸ケーブル、光ファイバーケーブル、ツイストペア、デジタル加入者回線 (DSL)、または赤外線、無線、およびマイクロ波などのワイヤレス技術を使用して、ウェブサイト、サーバ、または他のリモートソースから送信される場合、同軸ケーブル、光ファイバーケーブル、ツイストペア、DSL、または赤外線、無線、およびマイクロ波などのワイヤレス技術は、

40

50

媒体の定義に含まれる。ただし、コンピュータ可読記憶媒体およびデータ記憶媒体は、接続、搬送波、信号、または他の一時的媒体を含まないが、代わりに非一時的有形記憶媒体を対象とすることを理解されたい。本明細書で使用されるディスク (disk) およびディスク (disc) は、コンパクトディスク (disc) (CD)、レーザーディスク (登録商標) (disc)、光ディスク (disc)、デジタル多用途ディスク (disc) (DVD)、フロッピー (登録商標) ディスク (disk) および Blu-ray ディスク (disc) を含み、ここで、ディスク (disk) は、通常、データを磁気的に再生し、ディスク (disc) は、データをレーザーで光学的に再生する。上記の組合せもコンピュータ可読媒体の範囲内に含まれるべきである。

【0290】

10

[0262] 命令は、1つまたは複数のデジタル信号プロセッサ (DSP)、汎用マイクロプロセッサ、特定用途向け集積回路 (ASIC)、フィールドプログラマブル論理アレイ (FPGA)、あるいは他の等価な集積回路またはディスクリート論理回路など、1つまたは複数のプロセッサによって実行され得る。したがって、本明細書で使用される「プロセッサ」という用語は、上記の構造、または本明細書で説明された技法の実装に好適な他の構造のいずれかを指すことがある。さらに、いくつかの態様では、本明細書で説明された機能は、符号化および復号のために構成された専用ハードウェアおよび/またはソフトウェアモジュール内に与えられるか、あるいは複合コーデックに組み込まれ得る。また、本技法は、1つまたは複数の回路または論理要素で十分に実装され得る。

【0291】

20

[0263] 本開示の技法は、ワイヤレスハンドセット、集積回路 (IC : integrated circuit) またはICのセット (たとえば、チップセット) を含む、多種多様なデバイスまたは装置で実装され得る。本開示では、開示される技法を実行するように構成されたデバイスの機能的態様を強調するために、様々な構成要素、モジュール、またはユニットが説明されたが、それらの構成要素、モジュール、またはユニットは、必ずしも異なるハードウェアユニットによる実現を必要とするとは限らない。むしろ、上記で説明されたように、様々なユニットが、好適なソフトウェアおよび/またはファームウェアとともに、上記で説明された1つまたは複数のプロセッサを含めて、コーデックハードウェアユニットにおいて組み合わせられるか、または相互動作可能なハードウェアユニットの集合によって与えられ得る。

【0292】

30

[0264] 様々な例が説明された。これらおよび他の例は以下の特許請求の範囲内に入る。

以下に本願の出願当初の特許請求の範囲に記載された発明を付記する。

[C1]

ビデオデータをエントロピーコーディングする方法であって、前記方法は、前記ビデオデータのスライス中のシンタックス要素のための値をエントロピーコーディングするために、コンテキスト適応型エントロピーコーディングプロセスにおいて使用される複数のコンテキストのうちのコンテキストのためのあらかじめ定義された初期化値を取得することと、ここにおいて、前記あらかじめ定義された初期化値がNビット精度で記憶される、

前記あらかじめ定義された初期化値に基づいて、前記ビデオデータのスライスのための前記コンテキストの初期確率状態を決定することと、ここにおいて、前記コンテキストのための可能な確率状態の数が2のN乗よりも大きい、

前記コンテキストの前記初期確率状態に基づいて、前記シンタックス要素のための前記値のビンをエントロピーコーディングすることとを備える、方法。

[C2]

前記あらかじめ定義された初期化値に基づいて、傾斜値とオフセット値とを決定することをさらに備え、ここにおいて、前記初期確率状態を決定することが、

40

50

前記傾斜値と、前記オフセット値と、前記ビデオデータの前記スライスの量子化パラメータとに基づいて中間値を決定することと、
ルックアップテーブルを使用して、前記中間値を前記初期確率状態にマッピングすることと
を備える、C 1 に記載の方法。

[C 3]

前記ルックアップテーブルが、2 の N 乗よりも小さいかまたはそれに等しい数のエントリを含む、C 2 に記載の方法。

[C 4]

エントリの前記数が前記中間値の可能な値の数に等しい、C 3 に記載の方法。

10

[C 5]

前記コンテキストの前記初期確率状態を決定することが、ルックアップテーブルを使用して前記コンテキストの前記初期確率状態を決定することを備え、ここにおいて、前記ルックアップテーブル中の値が以下の式に従って定義され、

【数 2 9】

$$\text{MappedProb}[i] = \text{Ceil}(2^M * \text{prob}[i] + \text{オフセット})$$

上式で、`MappedProb[i]` は、前記ルックアップテーブル中の *i* 番目の値であり、`prob[i]` は、1 シンボルの可能な確率のセットを表すテーブル中の *i* 番目の値を表し、2 の *M* 乗は、前記コンテキストのための可能な確率状態の前記数を表し、`Ceil(x)` は、*x* よりも大きいかまたはそれに等しい最も小さい整数を示す関数である、C 1 に記載の方法。

20

[C 6]

`prob[i]` が 1 シンボルの *i* 番目の可能な確率を表す、C 5 に記載の方法。

[C 7]

前記コンテキストのための可能な確率状態の前記数が、2 の N 乗よりも大きいかまたはそれに等しい、C 5 に記載の方法。

[C 8]

N が 8 であり、前記コンテキストのための可能な確率状態の前記数が 2 の 15 乗である、C 5 に記載の方法。

30

[C 9]

オフセットが 0.5 または 0 に等しい、C 5 に記載の方法。

[C 10]

前記あらかじめ定義された初期化値に基づいて、傾斜値とオフセット値とを決定することをさらに備え、ここにおいて、前記初期確率状態を決定することが、

前記傾斜値と、前記オフセット値と、前記ビデオデータの前記スライスの量子化パラメータとに基づいて中間値を決定することと、

中間値と初期確率状態との間のマッピング関数を使用して、前記中間値に基づいて前記初期確率状態を決定することと

を備える、C 1 に記載の方法。

40

[C 11]

エントロピーコーディングすることが、前記ビンをエントロピー符号化することを備える、C 1 に記載の方法。

[C 12]

エントロピーコーディングすることが、前記ビンをエントロピー復号することを備える、C 1 に記載の方法。

[C 13]

前記コンテキスト適応型エントロピーコーディングプロセスが、コンテキスト適応型バイナリ算術コーディング (CABAC) プロセス、またはコンテキスト適応型可変長コーディング (CAVLC) プロセスを備える、C 1 に記載の方法。

50

[C 14]

ビデオデータのエントロピーコーディングのための装置であって、前記装置が、
前記ビデオデータのスライス中のシンタックス要素のための値をエントロピーコーディングするために、コンテキスト適応型エントロピーコーディングプロセスにおいて使用される複数のコンテキストを記憶するように構成されたメモリと、

1つまたは複数のプロセッサと
を備え、前記1つまたは複数のプロセッサは、

前記複数のコンテキストのうちのコンテキストのためのあらかじめ定義された初期化値を取得することと、ここにおいて、前記あらかじめ定義された初期化値がNビット精度で記憶される、

前記あらかじめ定義された初期化値に基づいて、前記ビデオデータのスライスのための前記コンテキストの初期確率状態を決定することと、ここにおいて、前記コンテキストのための可能な確率状態の数が2のN乗よりも大きい、

前記コンテキストの前記初期確率状態に基づいて、前記シンタックス要素のための前記値のビンをエントロピーコーディングすることと
を行うように構成された、装置。

[C 1 5]

前記1つまたは複数のプロセッサが、

前記あらかじめ定義された初期化値に基づいて、傾斜値とオフセット値とを決定するようさらに構成され、ここにおいて、前記初期確率状態を決定するために、前記1つまたは複数のプロセッサが、

前記傾斜値と、前記オフセット値と、前記ビデオデータの前記スライスの量子化パラメータとに基づいて中間値を決定することと、

ルックアップテーブルを使用して、前記中間値を前記初期確率状態にマッピングすることと

を行うように構成された、C 1 4 に記載の装置。

[C 1 6]

前記ルックアップテーブルが、2のN乗よりも小さいかまたはそれに等しい数のエントリを含む、C 1 5 に記載の装置。

[C 1 7]

エントリの前記数が前記中間値の可能な値の数に等しい、C 1 6 に記載の装置。

[C 1 8]

前記コンテキストの前記初期確率状態を決定するために、前記1つまたは複数のプロセッサが、ルックアップテーブルを使用するように構成され、ここにおいて、前記ルックアップテーブル中の値が以下の式に従って定義され、

【数30】

$$\text{MappedProb}[i] = \text{Ceil}(2^M * \text{prob}[i] + \text{オフセット})$$

上式で、`MappedProb[i]`は、前記ルックアップテーブル中の*i*番目の値であり、`prob[i]`は、1シンボルの可能な確率のセットを表すテーブル中の*i*番目の値を表し、2のM乗は、前記コンテキストのための可能な確率状態の前記数を表し、`Ceil(x)`は、*x*よりも大きいかまたはそれに等しい最も小さい整数を示す関数である、C 1 4 に記載の装置。

[C 1 9]

`prob[i]`が1シンボルの*i*番目の可能な確率を表す、C 1 8 に記載の装置。

[C 2 0]

前記コンテキストのための可能な確率状態の前記数が、2のN乗よりも大きいかまたはそれに等しい、C 1 8 に記載の装置。

[C 2 1]

Nが8であり、前記コンテキストのための可能な確率状態の前記数が2の15乗である、C 1 8 に記載の装置。

[C 2 2]

10

20

30

40

50

オフセットが 0 . 5 または 0 に等しい、 C 1 8 に記載の装置。

[C 2 3]

前記 1 つまたは複数のプロセッサが、

前記あらかじめ定義された初期化値に基づいて、傾斜値とオフセット値とを決定するようさらに構成され、ここにおいて、前記初期確率状態を決定するために、前記 1 つまたは複数のプロセッサが、

前記傾斜値と、前記オフセット値と、前記ビデオデータの前記スライスの量子化パラメータに基づいて中間値を決定することと、

中間値と初期確率状態との間のマッピング関数を使用して、前記中間値に基づいて前記初期確率状態を決定することと

を行うように構成された、 C 1 4 に記載の装置。

[C 2 4]

エントロピーコーディングするために、前記 1 つまたは複数のプロセッサが、前記ビンをエントロピー符号化するように構成された、 C 1 4 に記載の装置。

[C 2 5]

エントロピーコーディングするために、前記 1 つまたは複数のプロセッサが、前記ビンをエントロピー復号するように構成された、 C 1 4 に記載の装置。

[C 2 6]

前記コンテキスト適応型エントロピーコーディングプロセスが、コンテキスト適応型バイナリ算術コーディング (C A B A C) プロセス、またはコンテキスト適応型可変長コーディング (C A V L C) プロセスを備える、 C 1 4 に記載の装置。

[C 2 7]

前記装置が、

集積回路、

マイクロプロセッサ、または

ワイヤレス通信デバイス

のうちの少なくとも 1 つを備える、 C 1 4 に記載の装置。

[C 2 8]

復号ビデオデータを表示するように構成されたディスプレイをさらに備える、 C 2 7 に記載の装置。

[C 2 9]

前記ビデオデータをキャプチャするように構成されたカメラをさらに備える、 C 2 7 に記載の装置。

[C 3 0]

ビデオデータのエントロピーコーディングのための装置であって、前記装置は、

前記ビデオデータのスライス中のシンタックス要素のための値をエントロピーコーディングするために、コンテキスト適応型エントロピーコーディングプロセスにおいて使用される複数のコンテキストのうちのコンテキストのためのあらかじめ定義された初期化値を取得するための手段と、ここにおいて、前記あらかじめ定義された初期化値が N ビット精度で記憶される、

前記あらかじめ定義された初期化値に基づいて、前記ビデオデータのスライスのための前記コンテキストの初期確率状態を決定するための手段と、ここにおいて、前記コンテキストのための可能な確率状態の数が 2 の N 乗よりも大きい、

前記コンテキストの前記初期確率状態に基づいて、前記シンタックス要素のための前記値のビンをエントロピーコーディングするための手段とを備える、装置。

[C 3 1]

実行されたとき、ビデオコーディングデバイスの 1 つまたは複数のプロセッサに、

ビデオデータのスライス中のシンタックス要素のための値をエントロピーコーディングするために、コンテキスト適応型バイナリ算術コーディング (C A B A C) プロセスにお

10

20

30

40

50

いて使用される複数のコンテキストのうちのコンテキストのためのあらかじめ定義された初期化値を取得することと、ここにおいて、前記あらかじめ定義された初期化値がNビット精度で記憶される、

前記あらかじめ定義された初期化値に基づいて、前記ビデオデータのスライスのための前記コンテキストの初期確率状態を決定することと、ここにおいて、前記コンテキストのための可能な確率状態の数が2のN乗よりも大きい、

前記コンテキストの前記初期確率状態に基づいて、前記シンタックス要素のための前記値のビンをエントロピーコーディングすることと
を行わせる命令を記憶するコンピュータ可読記憶媒体。

【図1】

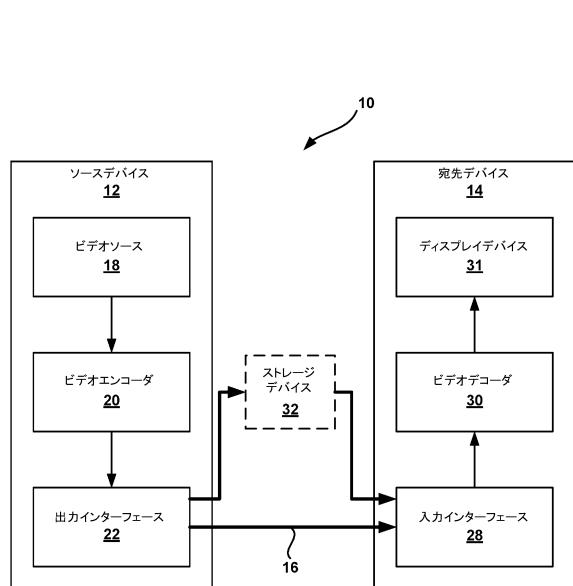


FIG. 1

【図2 A】

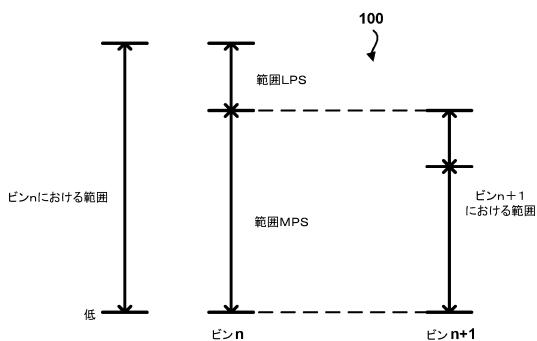


FIG. 2A

【図2 B】

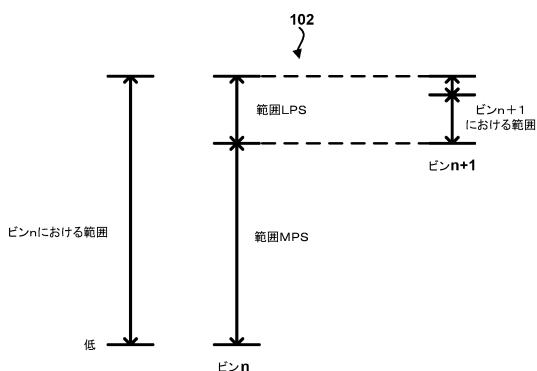


FIG. 2B

【 図 3 】

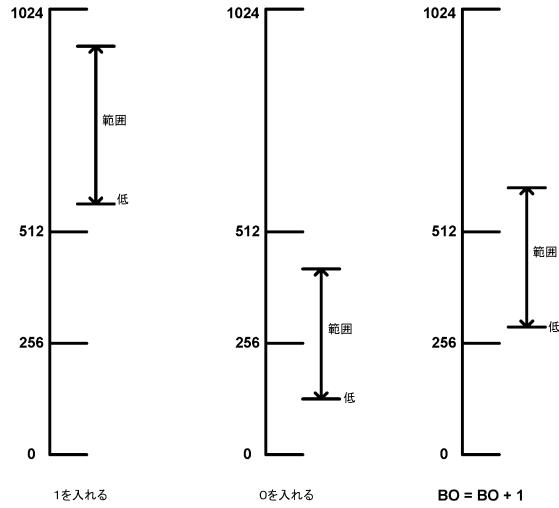


FIG. 3

【 図 4 】

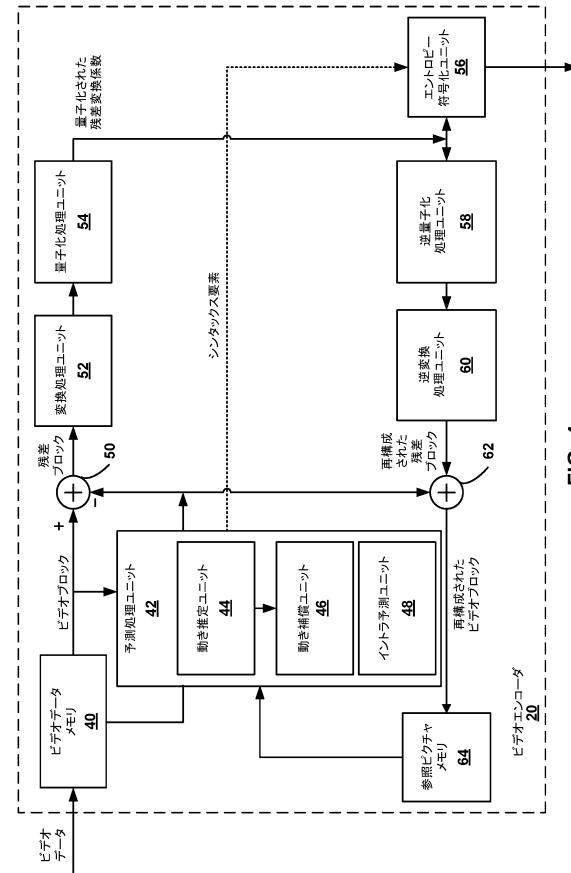


FIG. 4

【図5】

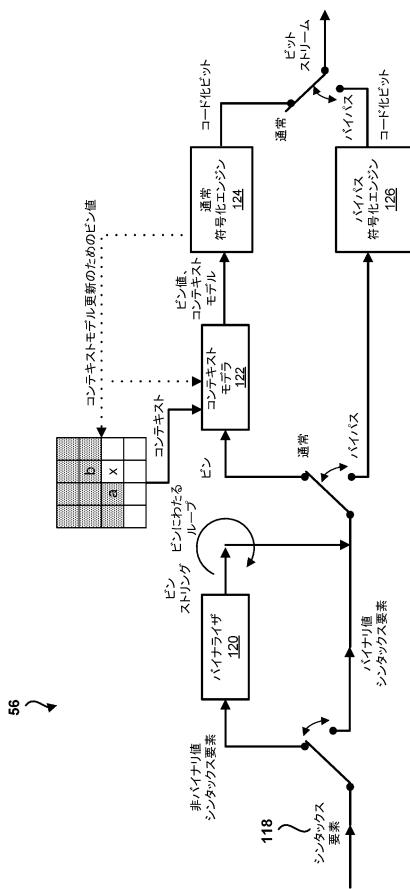
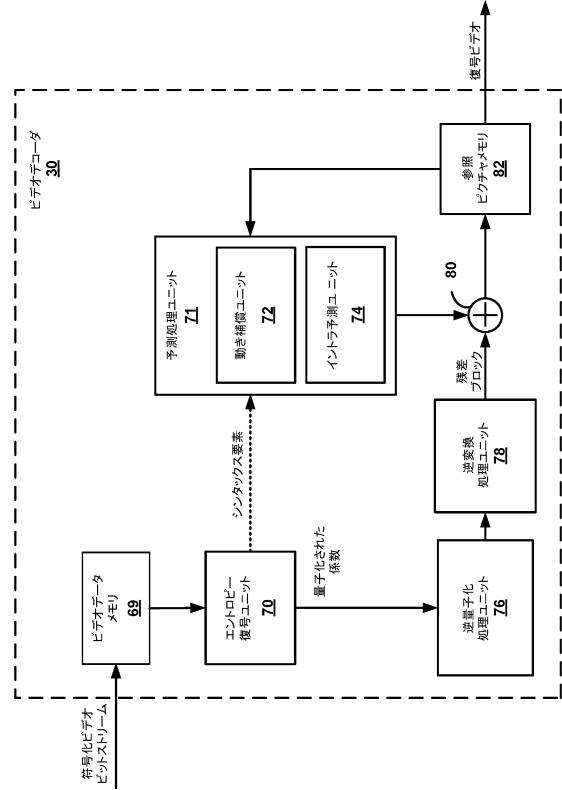


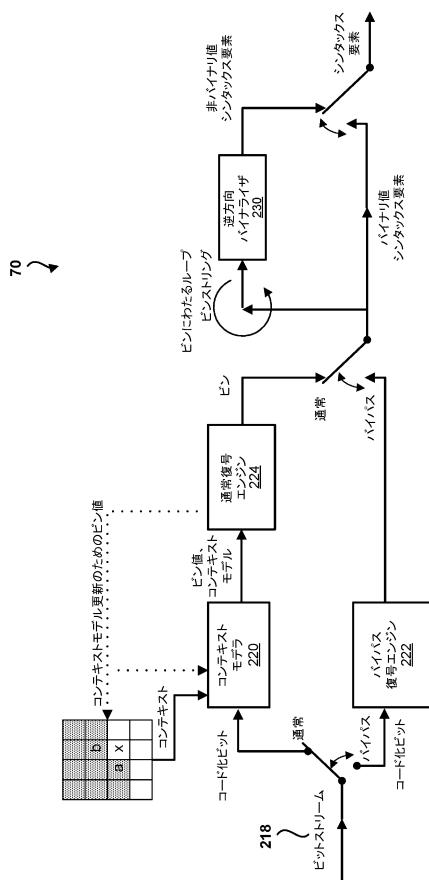
FIG. 5

【 四 6 】



۱۰۵

【図7】



【図8】

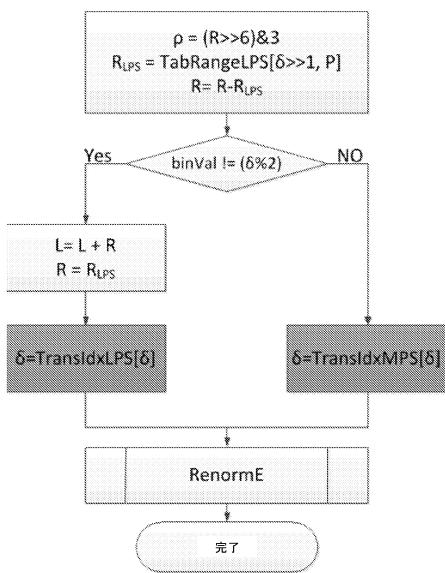
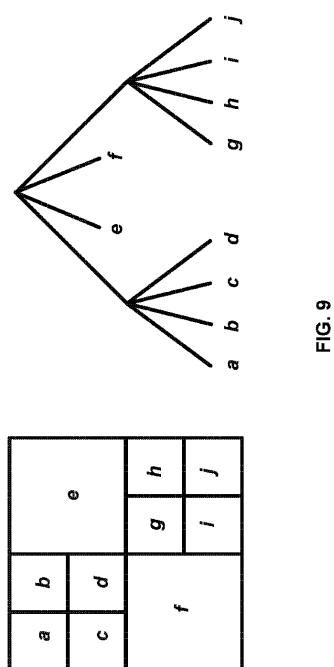


FIG. 8

【図9】



【図10】

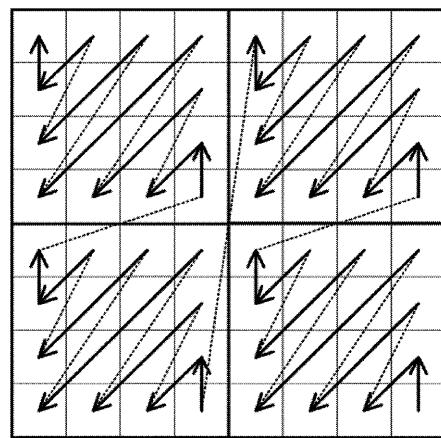
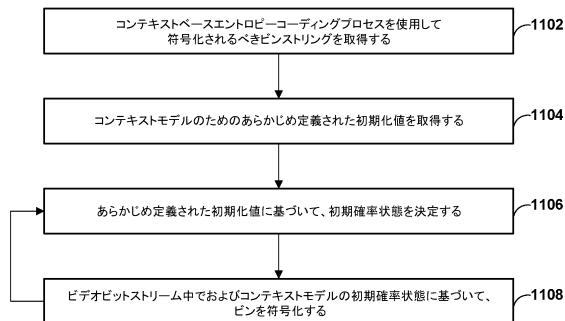
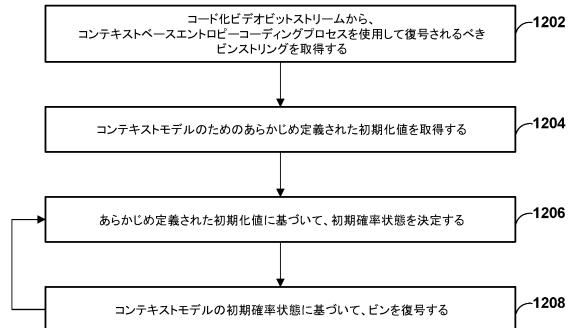


FIG. 10

【図11】



【図12】



フロントページの続き

- (72)発明者 ジャン、リ
アメリカ合衆国、カリフォルニア州 92121-1714、サン・ディエゴ、モアハウス・ドライブ 5775
- (72)発明者 チェン、ジャンレ
アメリカ合衆国、カリフォルニア州 92121-1714、サン・ディエゴ、モアハウス・ドライブ 5775
- (72)発明者 ジャオ、シン
アメリカ合衆国、カリフォルニア州 92121-1714、サン・ディエゴ、モアハウス・ドライブ 5775
- (72)発明者 リ、シャン
アメリカ合衆国、カリフォルニア州 92121-1714、サン・ディエゴ、モアハウス・ドライブ 5775
- (72)発明者 リウ、ホンビン
中華人民共和国、102208 ベイジン、チャンピン・ディストリクト、フイロングアン、ロンファユアン・セカンド・セクション、サーティーセブンス・ビルディング、ルーム 9-202
- (72)発明者 チェン、イン
アメリカ合衆国、カリフォルニア州 92121-1714、サン・ディエゴ、モアハウス・ドライブ 5775
- (72)発明者 カルチェビチ、マルタ
アメリカ合衆国、カリフォルニア州 92121-1714、サン・ディエゴ、モアハウス・ドライブ 5775

審査官 岩井 健二

- (56)参考文献 米国特許出願公開第2013/0114675(US, A1)
米国特許出願公開第2013/0107951(US, A1)
A.Alshin and E.Alshina, Multi-parameter probability up-date for CABAC, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, 6th Meeting: Torino, IT, 2011年 7月, JCTVC-F254_r4, pp.1-5
Alexander Alshin, Elena Alshina and JeongHoon Park, CE1 (subset B): Multi-parameter probability up-date for CABAC, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, 7th Meeting: Geneva, CH, 2011年11月, JCTVC-G764_r1, pp.1-4

- (58)調査した分野(Int.Cl., DB名)

H04N 19/00 - 19/98